LABELSTAR OFFICE

User Manual

Version 7.00 Build 1010
March 2021

Contents

Labelstar Office

Use this program to design and print your own labels.

The following program components are available:

- **Label Designing**

  With the Label Designer, labels can be created quick and easy.

- **Label Printing**

  For pure label printing, you can use Quick Print with a browser interface or Print Form with its "all in one" form and dynamic label preview.

- **Automated Printing**

  With the Folder Monitor, you can integrate label printing into your existing business application using a simple file interface. **Labelstar Office** has also a programming interface (API) that enables flexible application development.

Label Designer

With the **Label Designer** of **Labelstar Office** you can design and print your own labels.

- Simple operation by drag & drop
- Individual label design through various printer and system variables
- Support for all the most common barcode types
- Direct database connection possible
- Markups for flexible text formatting
- Print preview, logging, memory card support and other features

Variables

The purpose of variables is to insert certain changeable values on a label, for example the current date. This enables flexible label design.

```
$DateTime ("dd.MM.yyyy HH:mm", UpdateInterval=1, MonthOffset=10)
```

Labelstar Office supports the following types of variables:

- System Variables
- Printer Variables
- User-defined Variables

Reserved Characters

Certain characters within a printout signify and separate individual segments and permit a dismantling and processing of the printout.

The following table describes the reserved characters.

Character	Designation
$	Indicates the start of a variable.
(	Indicates the start of the parameter list.
)	Indicates the end of the parameter list.
"	Text identification
,	Parameter separator
=	Parameter value separator

Display a Dollar Sign

Whenever a character has a special meaning, problems result from it, since it is generally assumed that the function and not the character is meant. This means that when you enter a dollar sign, **Labelstar Office** first assumes that you want to define a variable - because a dollar sign is interpreted as the beginning of a variable name.

Now what if you really want to display a dollar sign?

Here's a very simple trick: double the dollar sign in order to display a dollar sign.

```
The item costs $DbField ("Item", "Price")$$.
```

Is output as follows:

```
The item costs 13$.
```


Syntax Conventions

The notation used to describe the syntax format is given below:

Convention	Usage
Bold	Indicates keywords, symbols, or punctuation that you must enter as shown.
Italics	Represents a parameter name for which you must supply a value.
[ ] brackets	Indicates an optional item.
{ } braces	Indicates an optional item that you can repeat as many times as appropriate.
| (vertical bar)	Indicates a list of mutually exclusive items (that is, you can select only one item from the list). Standard value is underlined.

System Variables

With the help of these variables, fields with variable contents can be defined on the label. In contrast to Printer Variables, system variables are managed and calculated by the application.

Supported Variable Types

- Date and Time Variables
- Counter
- User Input
- Field Management
- Database Variables
- Text Variables
- File Management
- Math Variables
- Misc Variables

Date and Time Variables

The date and time variables are a type of variable that allows you to insert information about the current system date and time into a field, the content of which is updated automatically when the date or time changes.

List of Available Date and Time Variables

	Variable	Description
Current Date	$CurrentDate	Returns the current system date.
Current Date/Time	$CurrentDateTime	Returns the current system date and time.
Current Time	$CurrentTime	Returns the current system time.
Date Difference	$DateDiff	Returns the number of days, months, or years between two dates.
Date/Time	$DateTime	Defines a system date and time variable.
Day of Week	$DayOfWeek	Returns a value that represents the day of the week for the specified date, where Sunday is the first day of the week.
Day of Year	$DayOfYear	Returns the day of the year (1 to 366) for the specified date.
Days in Month	$DaysInMonth	Returns the number of days in the month for the specified date.
Days in Year	$DaysInYear	Returns the number of days in the year for the specified date.
Format Date	$FormatDate	Returns an expression formatted as a date.
Now	$Now	Returns the current system date and time.
Parse Date	$ParseDate	Returns the parsed date according to the specified date format.
Today	$Today	Returns today's date.
Week of Year	$WeekOfYear	Returns the calendar week (1 to 53) for the specified date.

Current Date

Returns the current system date.

Syntax

```
$CurrentDate
```

Examples

System settings: 10/15/2014 2:03:59 PM

```
$CurrentDate -> "10/15/2014"
$FormatDate ($CurrentDate, "yyMMdd") -> "141015"
```

See also

- Current Date/Time
- Current Time
- Date/Time (System)
- Date/Time (Printer)

Current Date/Time

Returns the current system date and time.

Syntax

```
$CurrentDateTime
```

Examples

System settings: 10/15/2014 2:03:59 PM

```
$CurrentDateTime -> "10/15/2014 2:03:59 PM"
$FormatDate ($CurrentDateTime, "yyMMdd-HHmmss") -> "141015-140359"
```

See also

- Current Date
- Current Time
- Date/Time (System)
- Date/Time (Printer)

Current Time

Returns the current system time.

Syntax

```
$CurrentTime
```

Examples

System settings: 10/15/2014 2:03:59 PM

```
$CurrentTime -> "2:03:59 PM"
$FormatDate ($CurrentTime, "hhmmss") -> "020359"
$FormatDate ($CurrentTime, "HHmmss") -> "140359"
```

See also

- Current Date
- Current Date/Time
- Date/Time (System)
- Date/Time (Printer)

Date Difference

Returns the number of days, months, or years between two dates.

Syntax

```
$DateDiff (start_date, end_date, [Unit=unit], [Format=date_format], [Language=language])
```

Parameters

start_date
A date that represents the first, or starting, date.

end_date
A date that represents the last, or ending, date.

unit (optional, default = D)
The type of information to be returned.

Unit	Return Value
D	The number of days in the period.
M	The number of complete months in the period.
Y	The number of complete years in the period.
YD	The number of days in the period, ignoring the years of the dates.
YM	The number of complete months in the period, ignoring the years of the dates.

date_format (optional)
Defines the date format to be used to evaluate the start and end date. For more information, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

`$DateDiff ("6/1/2001", "8/15/2002", Unit="D")` -> 440
There are 440 days between June 1, 2001 and August 15, 2002.

`$DateDiff ("6/1/2001", "8/15/2002", Unit="YD")` -> 75
There are 75 days between June 1 and August 15, ignoring the years.

`$DateDiff ("1/1/2001", "1/1/2003", Unit="Y")` -> 2
There are two complete years between January 1, 2001 and January 1, 2003.

Today = "5/4/2020"

`$DateDiff ("1/1/2010", $Today, Unit="M")` -> 232
There are 124 complete months between January 1, 2010 and today (May 4, 2020).

Use different date formats

German date format
`$DateDiff ("01.06.2001", "05.08.2002", Unit="D", Language="de")` -> 440

Shortened date format
`$DateDiff ("010601", "020815", Unit="D", Format="yyMMdd")` -> 440

Date/Time

Defines a system date and time variable.

Syntax

```
$DateTime (output_format, [Prompt=prompt_text], [UpdateInterval=update_interval],
[MonthOffset=month_offset], [DayOffset=day_offset], [MinOffset=minute_offset], [StartDate=start_date],
[Language=language])
```

Parameters

output_format
Indicates how the date and time is to be formatted.

The output format can either be an individual format identifier (Standard Date and Time Format Strings) or a customized format pattern (Custom Date and Time Format Strings). If no output format is defined, the general format identifier 'G' is used.

prompt_text (optional)
If a prompt text is defined, the start date must be entered at print start.

update_interval (optional, default = 0)
Indicates how often the variable is to be updated during a print job.
0: At print start
1: After each label
n: After n labels
-1: At record change

month_offset (optional, default = 0)
Month offset (is added to the start date)

day_offset (optional, default = 0)
Day offset (is added to the start date)

minute_offset (optional, default = 0)
Minute offset (is added to the start time)

start_date (optional, as default the current date and time according to the system settings is used)
Defines the start date and time.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

System settings: 9/11/2013 1:20:35 PM

```
$DateTime ("d") -> "9/11/2013"
$DateTime ("d", Language = "de") -> "11.09.2013"
$DateTime ("d", Language = "en-GB") -> "11/09/2013"
$DateTime ("D", Language) -> "Wednesday, September 11, 2013"
$DateTime ("D", Language = "de") -> "Mittwoch, 11. September 2013"

$DateTime ("dd.MM.yyyy") -> "11.09.2013"
$DateTime ("dd.MM.yyyy", StartDate="6/12/2009", MonthOffset=2) -> "12.08.2009"
$DateTime ("D", UpdateInterval=0, Language="fr", StartDate=$ParseDate ("131012", "yyMMdd")) -> "samedi
12 octobre 2013"

$DateTime ("hh:mm:ss") -> "01:20:35"
$DateTime ("HH:mm:ss") -> "13:20:35"

ID01 = "260614"

$DateTime ("D", UpdateInterval=0, StartDate=$ParseDate (<<ID01>>, "ddMMyy")) -> "Thursday June 26, 2014"
$DateTime ("D", UpdateInterval=0, Language="es", StartDate=$ParseDate (<<ID01>>, "ddMMyy")) -> "jueves,
26 de junio de 2014"
```

Print the last day of the month

```
$DateTime ("l") -> "9/30/2013"
$DateTime ("l", Language="de") -> "30.09.2013"
$DateTime ("L") -> "Monday, September 30, 2013"
$DateTime ("L", Language="de", StartDate="13.02.2016") -> "Montag, 29. Februar 2016"
$DateTime ("dm.MM.yy", StartDate="02/13/2016") -> "29.02.16"
```

See also

- Date/Time (Printer)

Day of Week

Returns a value that represents the day of the week for the specified date, where Sunday is the first day of the week.

Syntax

```
$DayOfWeek (date, [Sunday=sunday], [Format=date_format], [Language=language])
```

Parameters

date
Any expression that represents a date.

sunday (optional, default = 0)
Specifies how the day of the week is to be returned.

date_format (optional)
Defines the format of the date expression. For details about valid format specifiers, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

Today = "8/15/2017" (Tuesday)

Return weekday as number

```
$DayOfWeek ($Today, Sunday=0) -> "2"
```

Example	Return Value
Sunday=0	Sunday = 0, Monday = 1, ... Saturday = 6
Sunday=1	Sunday = 1, Monday = 2, ... Saturday = 7
Sunday=10	Sunday = 10, Monday = 11, ... Saturday = 16

Return weekday as letter

```
$DayOfWeek ($Today, Sunday="A") -> "C"
```

Example	Return Value
Sunday="A"	Sunday = A, Monday= B, ... Saturday = G
Sunday="b"	Sunday = b, Monday = d, ... Saturday = h

Define a character (letter or digit) for each day of the week

`$DayOfWeek ($Today, Sunday="acegikm") -> "e"`

Example	Return Value
`Sunday="acegikm"`	Sunday = a, Monday = c, ... Saturday = m
`Sunday="7123456"`	Sunday = 7, Monday = 1, ... Saturday = 6

Note: The string must be at least seven characters long.

Define a string for each day of week

`$DayOfWeek ($Today, Sunday="black|white|yellow|red|blue|orange|green") -> "yellow"`

Example	Return Value
`Sunday="black|white|yellow|red|blue|orange|green"`	Sunday = black, Monday = white, ... Saturday = green

Note: The individual strings must be separated by a pipe character (|) and the list must contain at least seven strings.

See also

- Day of Year
- Days in Month
- Days in Year
- Week of Year

Day of Year

Returns the day of the year (1 to 366) for the specified date.

Syntax

```
$DayOfYear (date, [output_format], [Format=date_format], [Language=language])
```

Parameters

date
Any expression that represents a date.

output_format (optional, default = "o")
Defines the output format in which the day of the year is to be displayed on the label.

Format Specifier	Output
0	1-366
00	01-366
000	001-366
0000	0001-0366

For details about valid format specifiers, see Standard Numeric Format String or Custom Numeric Format Strings.

date_format (optional)
Defines the format of the date expression. For details about valid format specifiers, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

Today = "2/1/2014"

```
$DayOfYear ($Today) -> 32
$DayOfYear ($Today, "000") -> "032"
$DayOfYear ("8/15/2017") -> 227

$DayOfYear ("06/01/2001", Language="en-GB") -> 6
$DayOfYear ("01.06.2001", Language="de") -> 152

$DayOfYear ("020815", Format="yyMMdd") -> 227
```

See also

- Day of Week
- Days in Month
- Days in Year
- Week of Year

Days in Month

Returns the number of days in the month for the specified date.

Syntax

```
$DaysInMonth (date, [Format=date_format], [Language=language])
```

Parameters

date
Any expression that represents a date.

date_format (optional)
Defines the format of the date expression. For details about valid format specifiers, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

Today = "2/1/2014"

```
$DaysInMonth ($Today) -> 28
$DaysInMonth ("2/12/2016") -> 29 (Schaltjahr)
$DaysInMonth ("12/04/2016", Language="en-GB") -> 30
$DaysInMonth ("020815", Format="yyMMdd") -> 31
```

See also

- Day of Week
- Day in Year
- Days in Year
- Week of Year

Days in Year

Returns the number of days in the year for the specified date.

Syntax

```
$DaysInYear (date, [Format=date_format], [Language=language])
```

Parameters

date
Any expression that represents a date.

date_format (optional)
Defines the format of the date expression. For details about valid format specifiers, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

Today = "2/1/2014"

```
$DaysInYear ($Today) -> 365
$DaysInYear ("8/15/2016") -> 366
$DaysInYear ("15/08/2015", Language="en-GB") -> 365
$DaysInYear ("040815", Format="yyMMdd") -> 366 (leap year)
```

See also

- Day of Week
- Day of Year
- Days in Month
- Week of Year

Format Date

Returns an expression formatted as a date.

Syntax

```
$FormatDate (date, [output_format], [Language=language])
```

Parameters

date
Any expression that represents a date.

output_format (optional, default = "d")
Indicates how the date and time is to be formatted.

The output format can either be an individual format identifier (Standard Date and Time Format Strings) or a customized format pattern (Custom Date and Time Format Strings).

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

```
$FormatDate ("12.04.2016", "D") -> "Tuesday April 12, 2016"
$FormatDate ("12.04.2016", "D", Language="fr") -> "mardi 12 avril 2016"
$FormatDate ("13.04.2016", "D", Language="zh-CN") -> "2016年4月13日"
```

Today = "2/11/2019"

```
$FormatDate ($Today) -> "2/11/2019"
$FormatDate ($Today, Language="de") -> "11.02.2019"
```

See also

- Format Number
- Format Text

Now

Returns the current system date and time.

Syntax

```
$Now
```

Examples

System settings: 5/5/2020 1:43:08 PM

```
$Now -> "5/5/2020 1:43:08 PM"
$FormatDate ($Now, "yyMMdd-HHmmss") -> "200505-134308"
$FormatDate ($Now, "yyMMdd-hhmmss") -> "200505-014308"
```

See also

- Today

Parse Date

Returns the parsed date according to the specified date format.

The format of the string representation must match the specified date format exactly. Otherwise an error occured.

Syntax

```
$ParseDate (date_string, [date_format], [Language=language])
```

Parameters

date_string
Any expression to be converted to a date.

date_format (optional)
Defines the format of the date expression. For details about valid format specifiers, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

ID01 = "091210"

```
$ParseDate ("130910", "yyMMdd") -> "9/10/2013 12:00:00 AM"
$ParseDate ("1309101400", "yyMMddHHmm") -> "9/10/2013 2:00:00 PM"
$ParseDate (<<ID01>>, "yyMMdd") -> "12/10/2009 12:00:00 AM"
$ParseDate (<<ID01>>, "MMyydd") -> "9/10/2012 12:00:00 AM"
$ParseDate ("123456", "yyMMdd") ->  Error (The DateTime represented by the string is not supported in calendar.)
$ParseDate ("0912", "yyMMdd") -> Error (The string was not recognized as a valid DateTime.)

$ParseDate ("04/10/2012") -> "4/10/2012 12:00:00 AM"
$ParseDate ("04/10/2012", Language="en-GB") -> "10/4/2012 12:00:00 AM"
$ParseDate ("10.04.2012", Language="de") -> "4/10/2012 12:00:00 AM"
```


Today

Returns today's date.

Syntax

```
$Today
```

Examples

System settings: 5/5/2020 1:43:08 PM

```
$Today -> "5/5/2020"
$FormatDate ($Today, "yyMMdd") -> "200505"
```

See also

- Now

Week of Year

Returns the calendar week (1 to 53) for the specified date.

Note

There are country-specific deviations when calculating the calendar weeks. For more information, see Week Numbering Systems.

Syntax

```
$WeekOfYear (date, [output_format], [Format=date_format], [Language=language])
```

Parameters

date

Any expression that represents a date.

output_format (optional, Standard = "o")

Defines the output format in which the calendare week is to be displayed on the label.

Format Specifier	Output
0	1-53
00	01-53
000	001-053
YYYY	Year
ww	Calendare week (two digits)

For details about valid format specifiers, see Standard Numeric Format String or Custom Numeric Format Strings.

date_format (optional)

Defines the format of the date expression. For details about valid format specifiers, see Custom Date and Time Format Strings.

language (optional, as default the language set under Windows is used)

Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

Today = "2/1/2014"

```
$WeekOfYear ($Today) -> 5
$WeekOfYear ($Today, "00") -> "05"
$WeekOfYear ($Today, "YYYY-Www") -> "2014-W05"

$WeekOfYear ("12/31/2009", "YYYY-Www") -> "2009-W53"
$WeekOfYear ("12/31/2009", "YYYY-Www") -> "2013-W01"
```



```
$WeekOfYear ("3/4/2016") -> 10
$WeekOfYear ("03/04/2016", Language="en-GB") -> 13
$WeekOfYear ("04/03/2016", Language="en-GB") -> 9
$WeekOfYear ("04.03.2016", Language="de") -> 9
```

Date = "1/1/2011" (Saturday)

```
$WeekOfYear ("110101", "YYYY-Www", Format="yyMMdd") -> "2011-W01"
$WeekOfYear ("110101", "YYYY-Www", Format="yyMMdd", Language="de") -> "2010-W52"
```

See also

- Day of Week
- Day of Year
- Days in Month
- Days in Year

Week Numbering Systems

What is the Calendar Week?

Calendar weeks are the numbered weeks of a year. Altogether, the year usually has 52 calendar weeks.

There are few week numbering systems used around the world. Depending on the rule applied, the first week of the year can be set differently.

System 1

Most of European and Asian countries use a standardized ISO 8601 week date system in which the week with the number 1 is the first week which contains the first Thursday of January. In this system the first day of the week is Monday. It may happen that the first week of the year contains a few last days from the previous year. For example, the first week of the 2014 starts on December 30, 2013 (Monday) and ends on January 5, 2014 (Sunday).

System 2

USA, Canada, most of Latin America, Japan, Israel, South Korea, among others, use a week numbering system in which the first week (numbered 1) of any given year is the week which contains January 1st. The first day of a week is Sunday and Saturday is the last.

See also

- Week of Year

Counter

Defines a system counter.

Syntax

```
$Counter (start_value, [Prompt=prompt_text], [UpdateInterval=update_interval], [Increment=increment],
[MinValue=min_value], [MaxValue=max_value], [Radix=radix], [Mode=mode], [TrimLeft=trim_left])
```

Parameters

start_value
Current start value.
Note: The number of digits of the start value determines the output format. A maximum of 18 digits can be displayed.

prompt_text (optional, default = empty string)
If a prompt text is defined, the start value must be entered at print start.

update_interval (optional, default = 1)
Indicates how often the variable is to be updated during a print job.
1: After each label
n: After n labels
-1: At record change

increment (optional, Standard = 1)
The value to increment or decrement the counter.

min_value (optional)
Minimum value. If omitted the number of characters of the start value is used to calculate a minimum value.

Start Value	Radix	Calculated Min Value
0001	10	0000
001A	16	0000
ABC	1	AAA

max_value (optional)
Maximum value. If omitted the number of characters of the start value is used to calculate a maximum value.

Start Calue	Radix	Calculated Max Value
0001	10	9999
001A	16	FFFF
ABC	1	ZZZ

radix (optional, default = 10)
Radix, basis of counter
1: Alphabetical (A-Z)
2: Binary (0, 1)
8: Octal (0-7)
10: Decimal (0-9)
16: Hexadecimal (0-9, A-F)
36: Alphanumeric (0-9, A-Z)

mode (optional, default = 3)
Operating mode
0: Keep start value
1: Keep start value (automatic rollover)
2: Increment start value
3: Increment start value (automatic rollover)

trim_left (optional, default = 0)
Specifies whether leading zeros should be displayed or not.
0: Show leading zeros
1: Remove leading zeros

Examples

`$Counter ("0001", MinValue="0000", MaxValue="0009", Increment=1, Radix=10)` -> "0001, 0002, 0003, 0004, 0005, 0006, 0007, 0008, 0009, 0001, 0002, 0003, 0004, ..."
`$Counter ("0001", Increment=1, TrimLeft=1, Radix=10)` -> "1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ..."
`$Counter ("0005", MinValue="0000", MaxValue="0009", Increment=-1, Radix=10)` -> "0005, 0004, 0003, 0002, 0001, 0000, 0009, 0008, 0007, ..."
`$Counter ("0005", MinValue="0000", MaxValue="0009", Increment=1, Radix=10)` -> "0005. 0006, 0007, 0008, 0009, 0000, 0001, 0002, 0003, ..."

Number of copies = 200

`$Counter (`$Copies`, Increment=-1, Radix=10)` -> "200, 199, 198, 197, 196, 195, 194, 193, 192, 191, 190, ..."

Hexadecimal counter

`$Counter ("0009", MinValue="0000", MaxValue="FFFF", Increment=1, Radix=16)` -> "0009, 000A, 000B, 000C, 000D, 000E, 000F, 0010, 0011, 0012, ..."

See also

- Global Counter
- Counter (Printer)
- Numeric Counter (Printer)

Global Counter

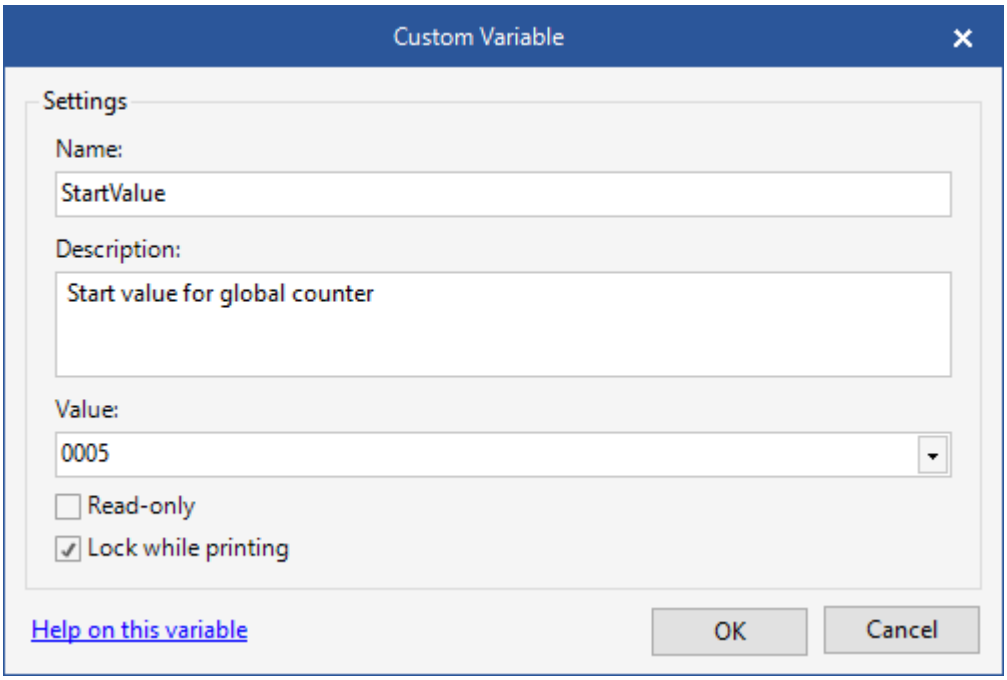
The **global counter** is a special case of a Counter (System). The start value becomes global, i.e. defined and saved across labels.

Example

A sample on how to create and print labels with a global counter can be found in the installation folder in the *Samples \Global Counter* directory or can be downloaded here.

To define a global counter, please proceed as follows:

1. Define a new user-defined variable.



2. Add a new text or barcode field on the label or select an existing field and open the Text or Data Editor.
3. Define a Counter (System) and insert as start value the user-defined variable.

```
$Counter ($StartValue)
```

User Input

Defines a system user input.

Example

A sample on how to simplify printing by using checkboxes or drop-down lists or on how to verify a user input by using a simple VBScript can be found in the installation folder in the *Samples\User Inputs* directory or can be downloaded here.

Syntax

```
$UserInput ([start_text])
```

Parameters

start_text (optional)
Text to be displayed when editing the user input. If omitted the start text defined in the internal parameters is used.

Internal Parameters

Prompt text
Prompt text that is displayed when editing the user input.

Update interval
Indicates how often the variable is to be updated during a print job.

Start text

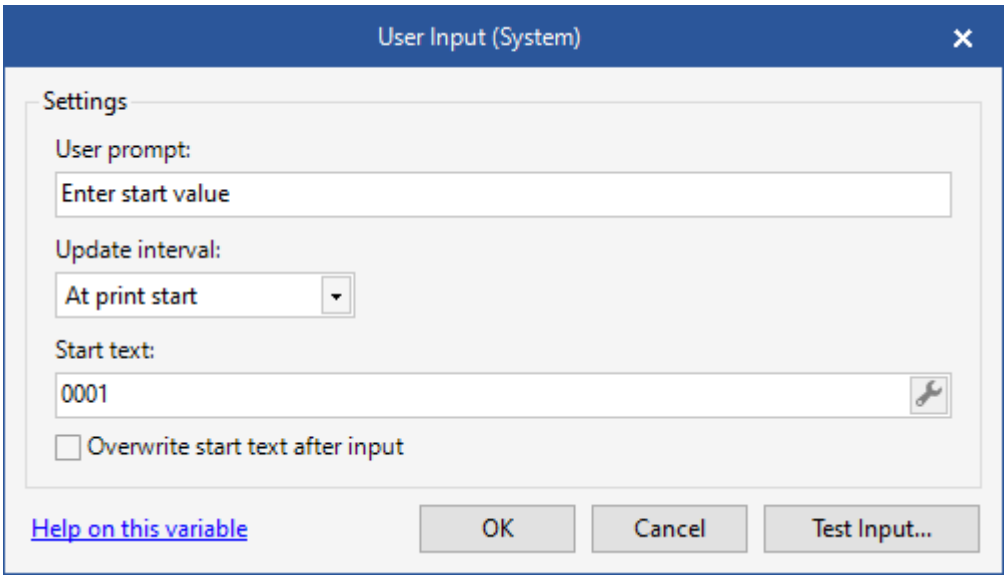
Text to be displayed when editing the user input. Click  to configure the settings of the input field.

The following table describes the available input types:

Type	Description
Text Field	Enables users to enter text.
Date Picker	Enables users to select a date value.
Checkbox	Enables users to choose between two options.
Drop-Down List	Enables users to pick from one ore more predefined values.
Slider	Enables users to pick a number from a range.

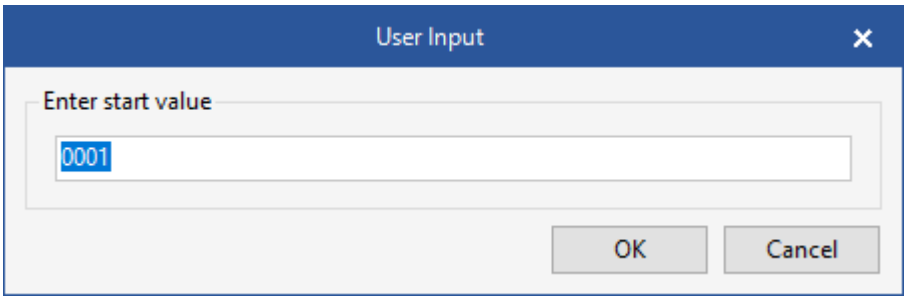
Overwrite start text after input
If this option is enabled text entered by the user overwrites existing start text; otherwise, the current start text is maintained.

Examples (Text Field)



User input

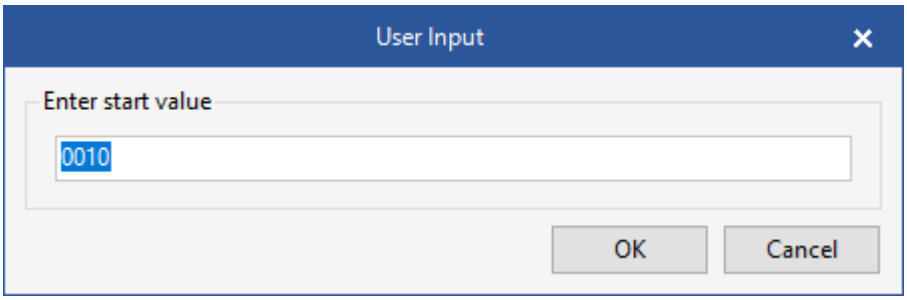
```
$UserInput ()
```



User input (start text = field link)

Text1 = "0010"

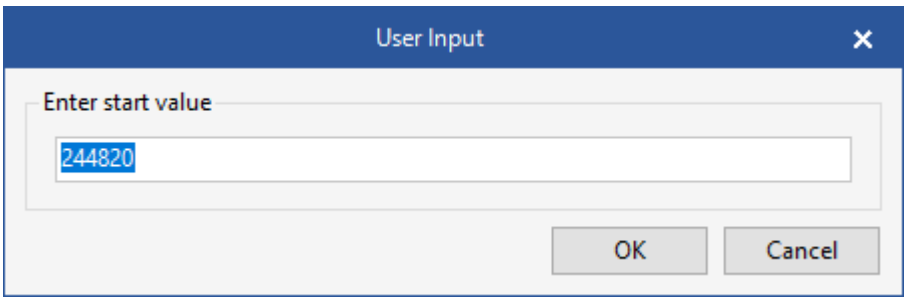
```
$UserInput (<<Text1>>)
```



User input (start text = database field)

	ID	Name	Capital	Area	Population	NativeName	Flag
▶	12	United Kingdom	London	244820	60440000	United Kingdom	

```
$UserInput ($DbField ("Europe", "Area"))
```



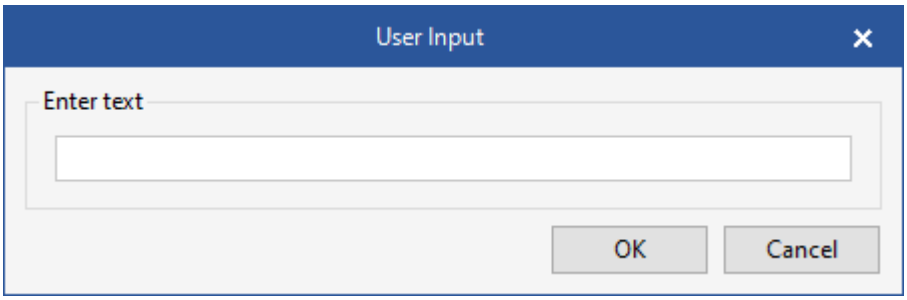
See also

- User Input (Printer)

Text Field

The **text field** enables the user to enter text. It is a basic input field with many different possible uses.

Primarily, **text fields** are used to enter free text, but can be provided with extensive formatting and validation options. As a result, a wide range of application specific inputs can be mapped using **text fields**.

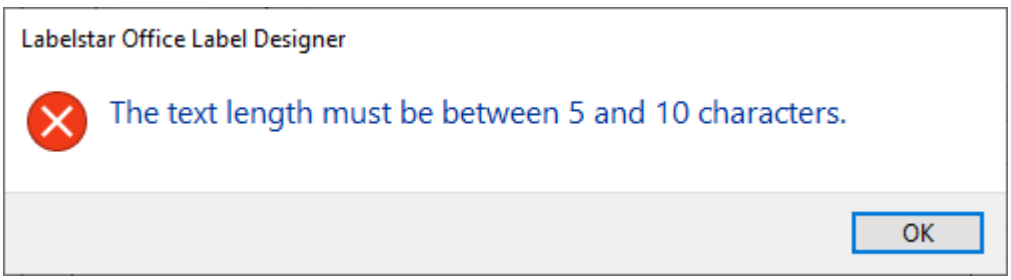


Configure Text Field

1. Add a new text or barcode field on the label or select an existing field and open the Text or Data Editor.
2. Create a new variable User Input (System) or select an existing variable $UserInput and click **Edit Variable**.
   The **User Input (System)** dialog box opens.
3. Click  to configure the settings of the text field.
   The **Configure Input Control** dialog box opens.
4. Select **Text Field** as input type.

Allowable Characters	Validation Options
Any characters Numbers only Letters only Numbers and letters	**Min length** Indicates the minimum number of characters that can be entered in the text box. **Max length** Indicates the maximum number of characters that can be entered in the text box.
Input mask	**Input mask** You can use the input mask to define a specific input format for the user input. For more information, see Input Format Strings.
Custom	**Character list** List of valid characters that can be entered in the text box. **Min length** Indicates the minimum number of characters that can be entered in the text box. **Max length** Indicates the maximum number of characters that can be entered in the text box.

Schlägt die Validierung fehl, weil z.B. zuwenig Zeichen eingegeben worden sind, wird eine Fehlermeldung angezeigt:



See also

- Advanced Options
- Date Picker
- Checkbox
- Drop-Down List
- Slider

Advanced Options

The following table describes the advanced properties for text fields:

Property	Description
AllowableCharacters	Determines which characters can be entered in the text box. • **All** - Any characters • **Numeric** - Numbers only • **Alpha** - Letters only • **Alphanumeric** - Numbers and letters • **InputMask** - Input mask, see InputMask Property • **Custom** - User-defined characters, see CustomCharacters Property
CustomCharacters	List of valid characters (*AllowableCharacters* = *Custom*)
Font	Specifies the font in which the text should be displayed. If no font is defined, the standard Windows font is used.
InputMask	Input mask (*AllowableCharacters* = *InputMask*) For more information, see Input Format Strings.
MaxLength	Indicates the maximum number of characters that can be entered in the text box.
MinLength	Indicates the minimum number of characters that can be entered in the text box.
RemoveLiterals	Deletes all literals from the input string before the text is processed. (*AllowableCharacters* = *InputMask*)
RemoveSpecialChars	Deletes all special characters from the input string before the text is processed. (*AllowableCharacters* = *InputMask*) Special characters are: point (.), colon (:), comma (,), semicolon (;), minus sign (-) and slash (/).

Input Format Strings

This format strings are used to define the input mask of a text field.

You can use input masks to ensure that data is always entered correctly. This can range from phone numbers to postal codes to serial numbers, and much more.

The following table describes the wildcard characters that can be used in the input mask:

Character	Description
0	The user must enter a number (0-9).
9	The user can enter a number (0-9).
#	The user can enter a number, a space, a plus or a minus sign.
L	The user must enter a letter.
?	The user can enter a letter.
A	The user must enter a letter or a number.
a	The user can enter a letter or a number.
&	The user must enter any character.
C	The user can enter any character.
>	Converts all characters thats follows to uppercase.
<	Converts all characters thats follows to lowercase.
|	Disables previous conversion to lowercase or uppercase.
\	The following characters are displayed literally.

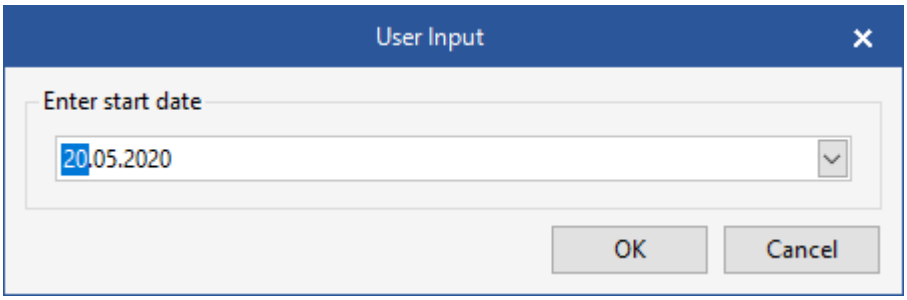
Examples of Inputs Masks

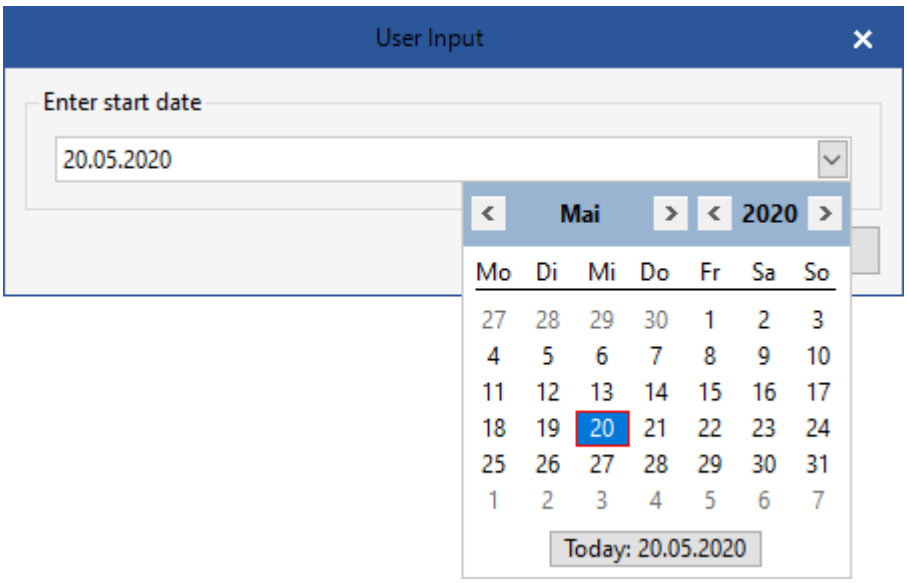
The examples in the following table illustrate some uses of input masks.

Input Mask	Usage	Input	Output
(000) 000-AAAA	Phone number (US format)	2065558353 206555TELE	(206) 555-8353 (206) 555-TELE
#999	Positive or negative number	-20 2000	-20 2000
ISBN 0-&&&&&&&&&-0	An ISBN number with a fixed text, mandatory first and last numbers and any combination of letters and numbers between these numbers.	155615-5077	ISBN 1-55615-507-7
>LL00000-0000	Article or serial number	db513920493	DB51392-0493

Date Picker

The **date picker** is an input field that is optimized to enter a single date or picking a single date from a calendar view.





Configure Date Picker

1. Add a new text or barcode field on the label or select an existing field and open the Text or Data Editor.
2. Create a new variable User Input (System) or select an existing variable $UserInput and click **Edit Variable**.
   The **User Input (System)** dialog box opens.
3. Click  to configure the settings of the text field.
   The **Configure Input Control** dialog box opens.
4. Select **Date Picker** as input type.
   - **Start date** Indicates which date is selected when the dialog box opens. By default, today's day is used as the start date.
   - **Earliest date** Defines the earliest date that can be entered.
   - **Latest date** Defines the latest date that can be entered.

See also

- Advanced Options
- Text Field
- Checkbox
- Drop-Down List
- Slider

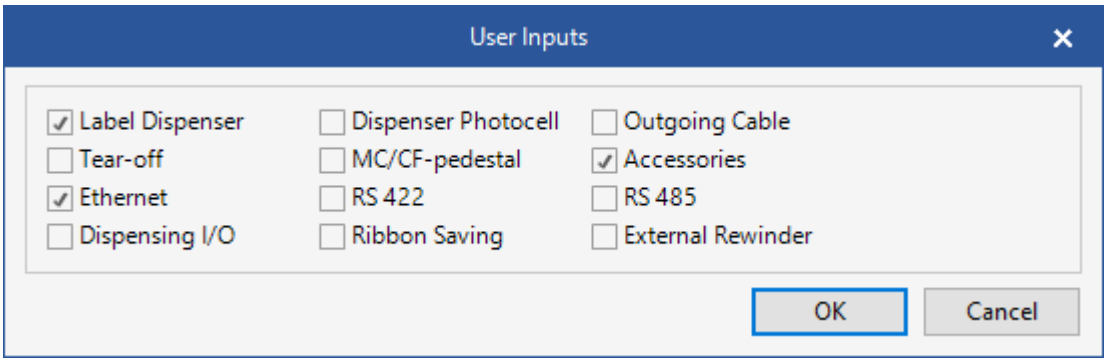
Advanced Options

The following table describes the advanced properties for date pickers:

Property	Description
Culture	Specifies culture-specific formatting information. As default the culture set under Windows is used.
CustomDateFormat	Sets the custom date/time format string. (*DateFormat* = *Custom*) For more information, see Standard Date and Time Format Strings and Custom Date and Time Format Strings.
DateFormat	Sets the standard date/time format string. *LongDate* Long date format (Wednesday, May 6, 2020) *ShortDate* Short date format (5/6/2020) *Custom* Custom date format, see CustomDateFormat Property
Font	Specifies the font in which the text should be displayed. If no font is defined, the standard Windows font is used.
MaxDate	Sets the maximum date that the user can enter. *MaxDate* has the following properties: • **DateTime** - Date value (*Mode* = *FixedDate*) • **DayOffset** - Day offset that is added to today's date. (*Mode* = *Today*) • **Mode** *DefaultDate* - 12/31/9998 *FixedDate* - Fixed date value, see DateTime Property *Today* - Variable date value based on the current system date • **MonthOffset** - Month offset that is added to today's date. (*Mode* = *Today*)
MinDate	Sets the minimum date the user can enter. *MinDate* has the following properties: • **DateTime** - Date value (*Mode* = *FixedDate*) • **DayOffset** - Day offset that is added to today's date. (*Mode* = *Today*) • **Mode** *DefaultDate* - 1/1/1753 *FixedDate* - Fixed date value, see DateTime Property *Today* - Variable date value based on the current system date • **MonthOffset** - Month offset that is added to today's date. (*Mode* = *Today*)
Preview	Preview for displaying the date on the label.
StartDate	Specifies the date displayed in the date picker. *StartDate* has teh following properties: • **DayOffset** - Day offset that is added to today's date. (*Mode* = *Today*) • **Mode** *FixedDate* - Fixed date value (= start text) *Today* - Variabler Datumswert, basierend auf dem aktuellen Systemdatum • **MonthOffset** - Month offset that is added to today's date. (*Mode* = *Today*)

Checkbox

The **checkbox** is an input field that enables the user to choose between two options (activated/not activated).



Is this the right control?

Use a checkbox whenever you want the user to make a yes/no choice.

Label the checkbox that it is true if the checkbox is activated and false if the checkbox is not activated.

Configure Checkbox

1. Add a new text or barcode field on the label or select an existing field and open the Text or Data Editor.
2. Create a new variable User Input (System) or select an existing variable $UserInput and click **Edit Variable**.
   The **User Input (System)** dialog box opens.
3. Click to configure the settings of the text field.
   The **Configure Input Control** dialog box opens.
4. Select **Checkbox** as input type.
   - **Value when checked** Value that is returned when checkbox is activated.
   - **Value when unchecked** Value that is returned when checkbox is not activated.

Example

A sample on how to simplify printing by using checkboxes can be found in the installation folder in the *Samples\User Inputs* directory or can be downloaded here.

See also

- Advanced Options
- Text Field
- Date Picker
- Drop-Down List
- Slider

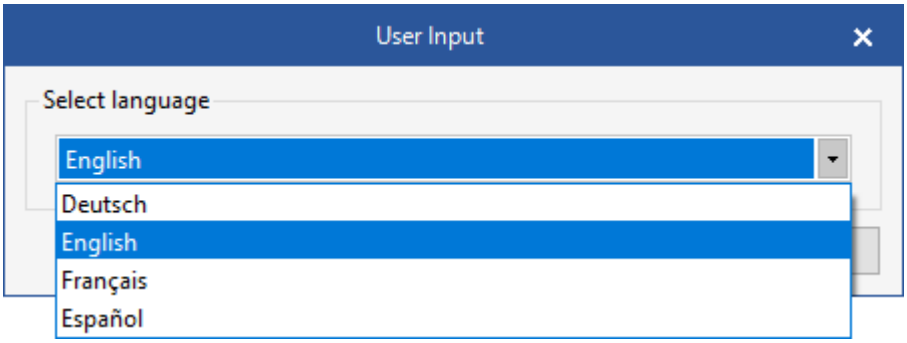
Advanced Options

The following table describes the advanced properties for checkboxes:

Property	Description
CheckedValue	Value that is returned when checkbox is activated.
Font	Specifies the font in which the text should be displayed. If no font is defined, the standard Windows font is used.
UncheckedValue	Value that is returned when checkbox is not activated.

Drop-Down List

The **drop-down list** is an input field that displays a list of input options from which the user can select one. The **drop-down list** starts in compact state and expands when you click on it to show a list of selectable items.



Is this the right control?

Use the drop-down list if you want to enable the user to select a single item from a set of items that can be clearly described by single lines of text.

Make sure that the items are arranged in a way that makes the most sense to the user.

Configure Drop-Down List

1. Add a new text or barcode field on the label or select an existing field and open the Text or Data Editor.
2. Create a new variable User Input (System) or select an existing variable $UserInput and click **Edit Variable**.
   The **User Input (System)** dialog box opens.
3. Click  to configure the settings of the text field.
   The **Configure Input Control** dialog box opens.
4. Select **Drop-Down List** as input type.
   - **Add new item** Click **Add** to insert a new item to the list. Enter the text to be displayed in the drop-down list and on the label.
     Note: Sometimes it makes sense to display a different text in the drop-down list than the value that should be displayed on the label. If e.g. the personnel number should be printed on the label, the corresponding employee names can be displayed in the drop-down list to make the selection easier for the user. Then enter the employee name as **Text** and the personnel number as **Value**.
   - **Edit item** Select an item and click  .
   - **Delete item** Select an item and click  .
   - **Move item** Select an item and click **Move up** or **Move down** to move the item up or down one position, or drag the selected item to the desired position within the list.
   - **Sort** Specifies the order in which the items should be displayed in the drop-down list.

Example

A sample on how to simplify printing by using drop-down lists can be found in the installation folder in the *Samples\User Inputs* directory or can be downloaded here.

See also

- Advanced Options
- Text Field
- Date Picker
- Checkbox
- Slider

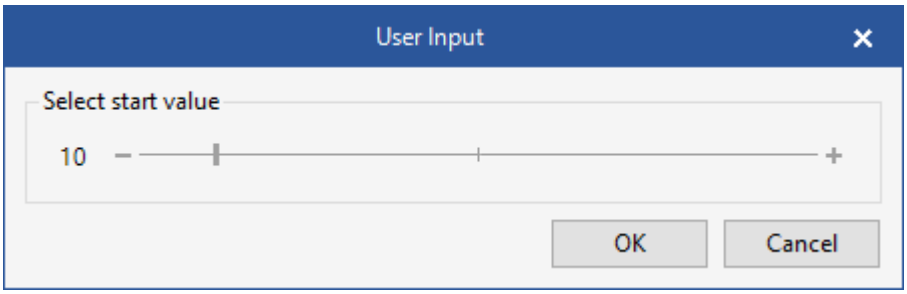
Advanced Options

The following table describes the advanced properties for drop-down lists:

Property	Description
Font	Specifies the font in which the text should be displayed. If no font is defined, the standard Windows font is used.
Items	Items to be displayed in the drop-down list. An item has the following properties: • **DisplayText** - Text that appears in the drop-down list. • **ValueText -** Text that appears on the label. If no *ValueText* is defined, *DisplayText* is used.
MaxDropDownItems	Defines the maximum number of items that are displayed in the drop-down list, without scolling.
SortType	Specifies the order in which the items should be displayed in the drop-down list. • **None** - The items are displayed as they were entered. • **Text** - The items are displayed in alphabetical order. • **TextNoCase** - The items are displayed in alphabetical order, ignore case. • **Numeric** - The items are displayed in numerical order.
WatermarkText	Text that is displayed if no item is selected. 

Slider

The **slider** is an input field that lets the user select from a range of values by moving a thumb along a track.



Is this the right control?

A slider is a good choice if you want to allow the user to select a value from a set of defined, related values.

Should the user enter an exact numerical value? If yes, it is better to use a numeric text field.

Configure Slider

1. Add a new text or barcode field on the label or select an existing field and open the Text or Data Editor.
2. Create a new variable User Input (System) or select an existing variable $UserInput and click **Edit Variable**.
   The **User Input (System)** dialog box opens.
3. Click  to configure the settings of the text field.
   The **Configure Input Control** dialog box opens.
4. Select **Slider** as input type.
   - **Min value** Sets the minimum selectable value for the slider.
   - **Max value** Sets the maximum selectable value for the slider.
   - **Increment** Sets the size of each movement (increment or decrement) of the slider.

See also

- Advanced Options
- Text Field
- Date Picker
- Checkbox
- Drop-Down List

Advanced Options

The following table describes the advanced properties for sliders:

Property	Description
DecreaseTooltip	Sets the text that is shown when the mouse hovers over the decrease (-) button of the slider.
DisplayFormat	Sets the format used to display the slider's value. *DisplayFormat* = "0\%"  *DisplayFormat* = "0 mm/s" 
Font	Specifies the font in which the text should be displayed. If no font is defined, the standard Windows font is used.
IncreaseTooltip	Sets the text that is shown when the mouse hovers over the increase (+) button of the slider.
Increment	Sets the size of each movement (an increment or jump between values) of the slider. This can be used together with *MinValue* and *MaxValue* to create a range of allowed values.
LabelWidth	Sets the width of the value label in pixels.
MaxValue	Sets the maximum selectable value for the slider.
MinValue	Sets the minimum selectable value for the slider.
ValueVisible	Determines whether the slider's value is displayed or not.

Field Management

With these variables it is possible to define fields that are composed of the content of several other fields.

List of Available Variables

	Variable	Description
Database Field	$DbField	Returns the contents of the specified database field.
Field Link	$FieldLink	Returns the content of the specified field.
Field Name	$FieldName	Returns the name of the current field.

Field Link

Returns the content of the specified field.

Syntax

```
$FieldLink (field_name, [placeholder_text])
```

```
<<field_name>>
```

Parameters

field_name
Field name

placeholder_text (optional)
Placeholder for on-screen display. The current field content is displayed by default.

Examples

ID01 = "12345"
ID02 = "abcABC"

```
$FieldLink (ID01) -> "12345"
$FieldLink (ID02) -> "abcABC"

<<ID01>> -> "12345"
<<ID02>> -> "abcABC"
```

See also

- Field Link (Printer)

Field Name

Returns the name of the current field.

Syntax

```
$FieldName
```


Database Variables

With these variables it is possible to read data from a database and use it on the label.

List of Available Database Variables

	Variable	Beschreibung
Database Field	$DbField	Returns the contents of the specified database field.
Database Path	$DbPath	Returns the full path of the database file.

Database Field

Returns the contents of the specified database field.

Syntax

```
$DbField (database_name, column_name, [NullValue=null_value], [Format=output_format])
```

Parameters

database_name
The name of the data connection.

column_name
The column name.
Note: Upper and lower case is considered.

null_value (optional)
Indicates which value is to be used if the appropriate database field is empty.

output_format (optional)
Indicates how the contents of the database field is to be formatted. For more information, see Format Strings.

Examples

	ID	Name	Capital	Area	Population	NativeName	Flag
▶	12	United Kingdom	London	244820	60440000	United Kingdom	

```
$DbField ("Europe", "Area") -> "244820"
$DbField ("Europe", "Area", Format="0000000000") -> "0000244820"
$DbField ("Europe", "Capital", Format="LLLL") -> "Lond"
$ToUpper ($DbField ("Europe", "Capital")) -> "LONDON"
```

Check whether a database field is empty

```
$If ($Length ($DBField (...)) = 0, "The database field is empty.", "The database field is not empty.")
$If ($IsEmpty ($DBField (...)), "The database field is empty.", "The database field is not empty.")
```

An example of how to create a database label with system variables can be found here.

See also

- Database Path
- Database Field (Printer)
- Datenbanken

Database Path

Returns the full path of the database file.

Syntax

```
$DbPath (database_name)
```

Parameters

database_name
The name of the data connection.

See also

- Database Field (System)
- Databases

Text Variables

These variables enables you to edit and evaluate text on the label.

List of Available Text Variables

	Variable	Description
Compare Strings	$Compare	Compares two strings lexicographically.
Contains String	$Contains	Returns a value indicating whether a specified substring occurs within another string.
Compare End of String	$EndsWith	Returns a value indicating whether the end of a string matches the specified substring.
Extract Substring	$Extract	Returns a substring from the specified string, using a delimiter string to devide the string into substrings.
Find String	$Find	Returns the position of the first occurrence of one string within another.
Find String Reverse	$FindRev	Returns the position of an occurance of one string within another, from the end of the string
Format Number	$FormatNumber	Returns an expression formatted as a number.
Format Text	$FormatText	Returns a formatted string by using the specified mask pattern.
HEX to String	$HexToString	Returns the string representation of a hex-encoded string.
Empty	$IsEmpty	Returns a value indicating whether a specified string is empty.
Equal	$IsEqual	Returns a value indicating whether two strings are equal.
Left	$Left	Returns a string containing a specified number of characters from the left side of the string.
Text Length	$Length	Returns the number of characters in the string.
Match Pattern	$MatchPattern	Returns a value indicating whether a specified regular expression finds a match within the string.
Middle	$Mid	Returns a string containing a specified number of characters.
Pad String from Left	$PadLeft	Returns a right-aligned string by adding spaces or a specified character to the left.
Pad String from Right	$PadRight	Returns a left-aligned string by adding spaces or a specified character to the right.
Remove Characters	$Remove	Returns a string without the specified range characters.
Replace Substring	$Replace	Returns a string where all occurrences of a specified substring has been replaced with another substring.
Replace Pattern	$ReplacePattern	Returns a string where all occurrences of a substring that matches a regular expression pattern has been replaced with another substring.
Reverse String	$Reverse	Returns a string in which the character order is reversed.
Right	$Right	Returns a string containing a specified number of characters from the right side of the string.
Compare Start of String	$StartsWith	Returns a value indicating whether the start of a string matches the specified substring.
String to HEX	$StringToHex	Returns a hex-encoded string.
Substring	$SubString	Returns a string containing a specified number of characters.
String to Lowercase	$ToLower	Returns a string that has been converted to lowercase.
String to Uppercase	$ToUpper	Returns a string that has been converted to uppercase.
Trim Spaces	$Trim	Returns a string without leading and trailing spaces, and replace multiple spaces with a single space.
Truncate String	$Truncate	Returns a string truncated to the specified length, and appends an optional trailing string if longer.

Compare Strings

Compares two strings lexicographically.

Syntax

```
$Compare (string1, string2, [IgnoreCase=ignore_case])
```

Parameters

string1
A string expression.

string2
A string expression.

ignore_case (optional, default = 0)
Indicates whether the comparison should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

Returns **0** if the strings are equal, **-1** if the first string is less than the second one, and **1** otherwise.

Examples

ID01 = "Word"
ID02 = "word"
ID03 = "w ord"

```
$Compare (<<ID01>>, "Word") -> 0
$Compare (<<ID02>>, "Word") -> -1
$Compare (<<ID02>>, "Word", IgnoreCase=1) -> 0
$Compare (<<ID03>>, "Word") -> -1
$Compare ("Word", <<ID03>>) -> 1
```

See also

- Empty
- Equal
- Compare Start of Strings
- Compare End of Strings
- Match Pattern

Contains String

Returns a value indicating whether a specified substring occurs within another string.

Syntax

```
$Contains (string1, string2, [IgnoreCase=ignore_case])
```

Parameters

string1
The string to be searched.

string2
The string to search for.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

Returns **1** if the substring is found, otherwise **0**.

Examples

```
$Contains ("This is a beautiful day!", "beautiful") -> 1
$Contains ("Jennifer", "an") -> 0
$Contains ("Sanderson", "an") -> 1
$Contains ("Andrew", "an") -> 0
$Contains ("Andrew", "an", IgnoreCase=1) -> 1
```

See also

- Find Text
- Find Text Reverse

Compare End of String

Returns a value indicating whether the end of a string matches the specified substring.

Syntax

```
$EndsWith (string1, string2, [IgnoreCase=ignore_case])
```

Parameters

string1
The string to be searched.

string2
The string to be matched with the end of *string1*.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

Returns **1** if the string ends with the specified substring, otherwise **0**.

Examples

```
$EndsWith ("This is a beautiful day!", "beautiful") -> 0
$EndsWith ("This is a beautiful day!", "day!") -> 1
$EndsWith ("Jennifer", "n") -> 0
$EndsWith ("Sanderson", "n") -> 1
$EndsWith ("Heidi", "n") -> 0
$EndsWith ("Coleen", "n") -> 1
```

See also

- Empty
- Equal
- Compare String
- Compare Start of String
- Match Pattern

Extract Substring

Returns a substring from the specified string, using a delimiter string to devide the string into substrings.

The string is interpreted as an alternating sequence of delimiters and tokens. So for the string "abc-defgh-i-jkl", where the delimiter character is "-", the tokens are "abc", "defgh", "i", and "jlk". Think of these as tokens 1 through 4. **$Extract** returns the token corresponding to the index number. When the index number is positive, tokens are counted starting from the left end of the string; when the index number is negative, tokens are counted starting from the right.

Syntax

```
$Extract (string, delimiter, index)
```

Parameters

string
The string to be searched.

delimiter
The string that delimits the substrings.

index
Index of the substring to be extracted. 1 is the first substring, 2 is the second substring and so on. You can count from the end of the string by using negative index.

Remarks

The function returns the requested substring, or an empty string (""), if the requested substring cannot be extracted.

Examples

```
$Extract ("a-b-c-d", "-", 2) -> "b"
$Extract ("a|b|c|d", "|", -2) -> "c"
$Extract ("a-b-c-d", "-", 10) -> ""
```

See also

- Left
- Middle
- Right

Find String

Returns the position of the first occurrence of one string within another.

Syntax

```
$Find (string1, string2, [StartIndex=start_index], [IgnoreCase=ignore_case])
```

Parameters

string1
The string to be searched.

string2
The string to search for.

start_index (optional, default = 1)
The position of the first character where the search should start. If no start index is specified, the search starts at the beginning of *string1*.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

If *string2* is found, the function returns a number that represents its position in *string1*. If *string2* is not found, the function returns **0**.

Examples

```
$Find ("This is a beautiful day!", "beautiful") -> 11
$Find ("This is a beautiful day!", "week") -> 0
$Find ("Sanderson", "n", StartIndex = 1) -> 3
$Find ("Sanderson", "n", StartIndex = 5) -> 9
```

See also

- Contains String
- Find String Reverse

Find String Reverse

Gibt die Position des ersten Zeichens einer Zeichenfolge innerhalb einer anderen Zeichenfolge, vom Ende der Zeichenfolge gesehen, zurück.

Syntax

```
$FindRev (string1, string2, [StartIndex=start_index], [IgnoreCase=ignore_case])
```

Parameters

string1
The string to be searched.

string2
The string to search for.

start_index (optional, default = 1)
The position of the first character where the search should start. If no start index is specified, the search starts at the end of *string1*.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

If *string2* is found, the function returns a number that represents its position in *string1*. If *string2* is not found, the function returns **0**.

Examples

```
$FindRev ("This is a beautiful day!", "beautiful") -> 11
$FindRev ("This is a beautiful day!", "week") -> 0
$FindRev ("Sanderson", "n", StartIndex = 1) -> 9
$FindRev ("Sanderson", "n", StartIndex = 5) -> 3
```

See also

- Contains String
- Find String

Format Number

Returns an expression formatted as a number.

Syntax

```
$FormatNumber (number, output_format, [Language=language])
```

Parameters

number
The number to be formatted.

output_format
Specifies how the number should be formatted. For more information, see Standard Numeric Format Strings or Custom Numeric Format Strings.

language (optional, as default the language set under windows is used)
Language which indicates which culture-specific format information is to be used. For more information, see Country Codes.

Examples

Show number with leading zeros

```
$FormatNumber (15, "00000") -> "00015"
$FormatNumber (-15, "00000") -> "-00015"
$FormatNumber (-15, "D5") -> "-00015"
```

Use different formatting for negative numbers and zero

Note: You can define special formats for negative numbers and zero. Use a semicolon ";" as separator in order to separate the formatting in two or three sections. The second section is for negative numbers, the third section is for zero.

```
$FormatNumber (15, "#;minus #;Null") -> "15"
$FormatNumber (-15, "#;minus #;Null") -> "minus 15"
$FormatNumber (0, "#;minus #;Null") -> "Null"
```

Use different languages

```
$FormatNumber (1234.56, "N2") -> "1.234,56"
$FormatNumber (1234.56, "N2", Language="en-US") -> "1,234.56"
$FormatNumber (1234.56, "N2", Language="fr-FR") -> "1 234,56"
```

See also

- Format Date
- Format Text

Format Text

Returns a formatted string by using the specified mask pattern.

Syntax

```
$FormatText (string, mask)
```

Parameters

string
The string to be formatted.

mask
Formatting pattern. For more information, see Text Format Strings.

Examples

```
$FormatString ("1234", "!0000-00") -> "0012-34"
$FormatString ("1234", "0000-00") -> "1234-00"
```

ProductCode	ProductName	Ingredients	Picture	Fat	SaturatedFat	TransFat
32500	Chocolate Coo...	organic pastry f...		7	3	0

```
$FormatString ($DbField ("Cookies", "ProductCode"), "!00.00.00") -> "03.25.00"
```

See also

- Format Date
- Format Number

HEX to String

Returns the string representation of a hex-encoded string.

Syntax

```
$HexToString (string)
```

Parameters

string
The HEX string to be converted.

Remarks

The hexadecimal string must contain an even number of HEX digits. A single hexadecimal value always consists of two digits and may only contain digits (0-9) and letters (a-f, A-F).

Examples

```
$HexToString ("3132333435") -> "12345"
$HexToString ("61626358595A") -> "abcXYZ"
$HexToString ("61626358595AX") -> Error (Invalid number of characters.)
$HexToString ("61626358595AXX") -> Fehler (Invalid hex digit at position 13.)
```

See also

- String to HEX

Empty

Returns a value indicating whether a specified string is empty.

Syntax

```
$IsEmpty (string)
```

Parameters

string
The string to be checked.

Remarks

Returns **1** if the string is empty, otherwise **0**.

Examples

```
$IsEmpty ("") -> 1
$IsEmpty ("This is a beautiful day!") -> 1
```

Check whether a database field is empty

```
$If ($IsEmpty ($DbField (...)), "The database field is empty.", "The database field is not empty.")
```

See also

- Equal
- Compare Strings
- Compare Start of String
- Compare End of String
- Match Pattern

Equal

Returns a value indicating whether two strings are equal.

Syntax

```
$IsEqual (string1, string2, [IgnoreCase=ignore_case])
```

Parameters

string1
String

string2
String to be compared with *string1*.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

Returns **1** if the strings are equal, otherwise **0**.

Examples

```
$IsEqual (12, 20) -> 0
$IsEqual (12, 12) -> 1
$IsEqual ("Jennifer", "Pablo") -> 0
$IsEqual ("Pablo", "pablo") -> 0
$IsEqual ("Pablo", "pablo", IgnoreCase=1) -> 1
$IsEqual ("Patricia", "Pablo") -> 0
$IsEqual ("Sanderson", "Pablo") -> 0
```

See also

- Empty
- Compare Strings
- Compare Start of String
- Compare End of String
- Match Pattern

Left

Returns a string containing a specified number of characters from the left side of the string.

Syntax

```
$Left (string, [length])
```

Parameters

string
The original string.

length (optional, default = 1)
Specifies how many characters to return. If set to 0, an empty string ("") is returned. If set to greater than or equal to the length of *string*, the entire string is returned.

Examples

```
$Left ("abcDEF") -> "a"
$Left ("abcDEF", 0) -> ""
$Left ("abcDEF", 2) -> "ab"
$Left ("abcDEF", 4) -> "abcD"
$Left ("abcDEF", 10) -> "abcDEF"
```

See also

- Middle
- Right
- Substring

Text Length

Returns the number of characters in the string.

Syntax

```
$Length (string)
```

Parameters

string
The original string.

Examples

```
$Length ("abcDEF") -> 6
$Length ("") -> 0
```

Check whether a database field is empty

```
$If ($Length ($DBField (...)) == 0, "The database field is empty.", "The database field is not empty.")
```

Match Pattern

Returns a value indicating whether a specified regular expression finds a match within the string.

Syntax

```
$MatchPattern (string, pattern, [IgnoreCase=ignore_case], [RightToLeft=right_to_left])
```

Parameters

string
The string to be searched.

pattern
Search pattern. Search patterns are regular expressions and are defined in a special syntax. This enables very complex searches. For more information, see Regular Expressions.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

right_to_left (optional, default = 0)
Specifies whether the search should be carried out from right to left or from left to right.
0: Pattern search is from left to right.
1: Pattern search is from right to left.

Remarks

Returns **1** if a match is found, otherwise **0**.

Examples

Search for first names that start with the letter 'P' and end with a vowel (a, e, i, o or u)

```
$MatchPattern ("Jennifer", "P.*[aeiou]$") -> 0
$MatchPattern ("Pablo", "P.*[aeiou]$") -> 1
$MatchPattern ("Patricia", "P.*[aeiou]$") -> 1
$MatchPattern ("Peggy", "P.*[aeiou]$") -> 0
```

Check whether certain words are contained in a text

ProductCode	ProductName	Ingredients	Picture
32500	Chocolate Cookie	organic pastry flour (organic whole grain white wheat), organic evaporated cane juice, organic butter (cream, salt), oganic dark chocolate chips (organic cacao mass, organic evaporated cane juice, organic cacao butter, may contain non-GMO soy lecithin), organic whole eggs, organic sunflower oil, organic vanilla extract, organic molasses, baking powder, baking soda, sea salt.	

```
$MatchPattern ($DbField ("Cookies", "Ingredients"), "\b(wheat|rye|barley|oats|soy|eggs)\b") -> 1
$MatchPattern ($DbField ("Cookies", "Ingredients"), "\b(almonds|hazelnuts|walnuts|cashews)\b") -> 0
```

See also

- Regular expressions
- Replace Pattern
- Empty
- Equal
- Compare Strings
- Compare Start of String
- Compare End of String

Middle

Returns a string containing a specified number of characters.

Syntax

```
$Mid (string, start_index, [length])
```

Parameters

string
The original string.

start_index
The position of the first character to be returned.

length (optional)
Specifies how many characters to return. If no value is assigned, all characters from the start position to the end of the string are returned.

Examples

```
$Mid ("abcDEF", 4) -> "DEF"
$Mid ("abcDEF", 4, 2) -> "DE"
$Mid ("abcDEF", 10, 2) -> ""
```

See also

- Left
- Right
- Substring

Pad String from Left

Returns a right-aligned string by adding spaces or a specified character to the left.

Syntax

```
$PadLeft (string, length, [PadChar=pad_char])
```

Parameters

string
The original string.

length
The desired string length.

pad_char (optional, default = space)
The character to pad *string* with, if the length of *string* is less than *length*.

Examples

```
$PadLeft ("abcDEF", 10, PadChar="X") -> "XXXXabcDEF"
$PadLeft ("abcDEF", 10) -> "    abcDEF"
$PadLeft ("12345", 10, PadChar="0") -> "0000012345"
```

See also

- Pad String from Right

Pad String from Right

Returns a left-aligned string by adding spaces or a specified character to the right.

Syntax

```
$PadRight (string, length, [PadChar=pad_char])
```

Parameters

string
The original string.

length
The desired string length.

pad_Char (optional)
The character to pad *string* with, if the length of *string* is less than *length*.

Examples

```
$PadRight ("abcDEF", 10, PadChar="X") -> "abcDEFXXXX"
$PadRight ("abcDEF", 10) -> "abcDEF    "
$PadRight ("12345", 10, PadChar="0") -> "1234500000"
```

See also

- Pad String from Left

Remove Characters

Returns a string without the specified range characters.

Syntax

```
$Remove (string, start_index, [length])
```

Parameters

string
The original string.

start_index
The position of the first character to be removed.

length (optional)
Specifies how many characters to delete. If no value is assigned, all characters from the start position to the end of the string are deleted.

Examples

```
$Remove ("abcDEF", 4) -> "abc"
$Remove ("abcDEF", 4, 2) -> "abcF"
```


Replace Substring

Returns a string where all occurrences of a specified substring has been replaced with another substring.

Syntax

```
$Replace (string, find, replace, [find, replace, ...], [IgnoreCase=ignore_case])
```

Parameters

string
The original string.

find
The string to be replaced.

replace
The string to replace each occurrence of *find*.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Examples

```
$Replace ("ABCDEFabcdef", "abc", "XXX") -> "ABCDEFXXXdef"
$Replace ("ABCDEFabcdef", "abc", "XXX", IgnoreCase = 1) -> "XXXDEFXXXdef"
```

Delete strings

```
$Replace ("ABCDEFabcdef", "ABC", "", "abc", "") -> "DEFdef"
$Replace ("ABCDEFabcdef", "abc", "", IgnoreCase = 1) -> "DEFdef"
$Replace ("1234-5678", "-", "") -> "12345678"
```

Convert numeric weekday to a letter

Today = "5/11/2020" (Monday)

```
$Replace ($DayOfWeek ($Today), "0", "A", "1", "B", "2", "C", "3", "D", "4", "E", "5", "F", "6", "G")
-> "B"
```

See also

- Replace Pattern
- Trim Spaces

Replace Pattern

Returns a string where all occurrences of a substring that matches a regular expression pattern has been replaced with another substring.

Syntax

```
$ReplacePattern (string, pattern, replace, [IgnoreCase=ignore_case], [RightToLeft=right_to_left])
```

```
$ReplacePattern (string, file_name, replace, [IgnoreCase=ignore_case], [RightToLeft=right_to_left], [MatchWholeWord=match_whole_word])
```

Parameters

string
The original string.

pattern
Search pattern. Search patterns are regular expressions and are defined in a special syntax. This enables very complex searches. For more information, see Regular Expressions.

file_name
The file name of the text file that contains the keywords to search for. The lines in the text file are linked by the OR operator to a regular expression. For an example, see Food Allergen Labelling.

replace
The string to replace each text pattern found.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

right_to_left (optional, default = 0)
Specifies whether the search should be carried out from right to left or from left to right.
0: Pattern search is from left to right.
1: Pattern search is from right to left.

match_whole_word (optional, is only taken into account when using a text file, default = 0)
Specifies whether the search should be performed for whole words only, or whether substring matches should be allowed.
0: Search matched partial as well as entire words.
1: Search matched entire words only.

Examples

`$ReplacePattern ("abcdefABCDEF", "abc|DEF", "<b>$0</b>")` -> "**abc**defABC**DEF**"
`$ReplacePattern ("abcdefABCDEF", "abc", "<u><b>$0</b></u>", IgnoreCase=true)` -> "**abc**def**ABC**DEF"

Delete invalid characters from string

In this case, `$ReplacePattern` strips out all nonalphanumeric characters except periods (.), at symbols (@), and hyphens (-), and returns the remaining string.

`$ReplacePattern ("<email>@example.com", "[^\w\.@-]", "")` -> "email@example.com"

Match whole words only

`$ReplacePattern ("The bobcat is a North American cat is about twice the size of the domestic cat.", "cat", "<b>$0</b>")` -> "The bob**cat** is a North American **cat** is about twice the size of the domestic **cat**."
`$ReplacePattern ("The bobcat is a North American cat is about twice the size of the domestic cat.", "\bcat\b", "<b>$0</b>")` -> "The bobcat is a North American **cat** is about twice the size of the domestic **cat**."

Replace multiple whitespaces with a single whitespace

`$ReplacePattern ("This  is  a  beautiful  day!", "\s+", " ")` -> "This is a beautiful day!"

See also

- Regular Expressions
- Match Pattern

Reverse String

Returns a string in which the character order is reversed.

Syntax

```
$Reverse (string)
```

Parameters

string
The original string.

Examples

```
$Reverse ("This is a beautiful day!") -> "!yad lufituaeb a si sihT"
$Reverse ("abcDEF") -> "FEDcba"
$Reverse ("12345") -> "54321"
```


Right

Returns a string containing a specified number of characters from the right side of the string.

Syntax

```
$Right (string, [length])
```

Parameters

string
The original string.

length (optional, default = 1)
Specifies how many characters to return. If set to 0, an empty string ("") is returned. If set to greater than or equal to the length of *string*, the entire string is returned.

Examples

```
$Right ("abcDEF") -> "F"
$Right ("abcDEF", 0) -> ""
$Right ("abcDEF", 2) -> "EF"
$Right ("abcDEF", 4) -> "cDEF"
$Right ("abcDEF", 10) -> "abcDEF"
```

See also

- Left
- Mid
- Substring

Compare Start of String

Returns a value indicating whether the start of a string matches the specified substring.

Syntax

```
$StartsWith (string1, string2, [IgnoreCase=ignore_case])
```

Parameters

string1
The string to be searched.

string2
The string to be matched with the start of *string1*.

ignore_case (optional, default = 0)
Indicates whether the search should care about lower and upper case.
0: Match case during comparison.
1: Ignore case during comparison.

Remarks

Returns **1** if the string starts with the specified substring, otherwise **0**.

Examples

```
$StartsWith ("This is a beautiful day!", "beautiful") -> 0
$StartsWith ("This is a beautiful day!", "This") -> 1
$StartsWith ("Pablo", "Pa") -> 1
$StartsWith ("Patricia", "Pa") -> 1
$StartsWith ("Sanderson", "Pa") -> 0
```

See also

- Empty
- Equal
- Compare Strings
- Compare End of String
- Match Pattern

String to HEX

Returns a hex-encoded string.

Syntax

```
$StringToHex (string)
```

Parameters

string
The ASCII string to be converted.

Remarks

Each ASCII character is converted to a two-digit hexadecimal value (with capital letters). Always returns an even number of HEX digits.

Examples

```
$StringToHex ("12345") -> "3132333435"
$StringToHex ("abcXYZ") -> "61626358595A"
```

See also

- HEX to String

Substring

Returns a string containing a specified number of characters.

Syntax

```
$Substring (string, start_index, [length])
```

Parameters

string
The original string.

start_index
The position of the first character to be returned.

length (optional)
Specifies how many characters to return. If no value is assigned, all characters from the start position to the end of the string are returned.

Examples

```
$Substring ("abcDEF", 4) -> "DEF"
$Substring ("abcDEF", 4, 2) -> "DE"
$Substring ("abcDEF", 10, 2) -> ""
```

See also

- Left
- Middle
- Right
- Substring (Printer)

String to Lowercase

Returns a string that has been converted to lowercase.

Syntax

```
$ToLower (string)
```

Parameters

string
The original string.

Examples

```
$ToLower ("abcDEF") -> "abcdef"
```

See also

- String to Uppercase

String to Uppercase

Returns a string that has been converted to uppercase.

Syntax

```
$ToUpper (string)
```

Parameters

string
The original string.

Examples

```
$ToUpper ("abcDEF") -> "ABCDEF"
```

See also

- String to Lowercase

Trim Spaces

Returns a string without leading and trailing spaces, and replace multiple spaces with a single space.

Syntax

```
$Trim (string)
```

Parameters

string
The original string.

Beispiele

`$Trim ("  Text  with  spaces  ")` -> "Text with spaces"

See also

- Replace Substring
- Replace Pattern

Truncate String

Returns a string truncated to the specified length, and appends an optional trailing string if longer.

Syntax

```
$Truncate (string, length, [Trailing=trailing_string])
```

Parameters

string
The original string.

length
Maximum number of characters, including ellipsis.

trailing_string (optional, default = "...")
Ellipsis append to the truncated string.

Examples

```
$Truncate ("Example text", 8) -> "Examp..."
$Truncate ("Example", 8) -> "Example"
```

Truncate Path

Application path = "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office\LabelDesigner.exe"

```
$Truncate ($AppPath, 40) -> "C:\Programs (x86)\...\LabelDesigner.exe"
```


File Management

With these variables it is possible to use file and directory information on the label.

List of Available Variables

	Variable	Description
Application Data Folder	$AppDataDir	Returns the full path to the directory containing application data for all users.
Application Folder	$AppDir	Returns the full path to the current application directory.
Application Path	$AppPath	Returns the full path to the current application file.
Folder Name	$Dir	Returns the directory information for the specified path string.
File Extension	$Ext	Returns the file extension of the specified path string.
File Name	$FileName	Returns the file name of the specified path string.
Image Folder	$ImageDir	Returns the full path to the default image directory.
Installation Folder	$InstallDir	Returns the full path to the directory in which **Labelstar Office** is installed.
Label Folder	$LabelDir	Returns the full path to the default label directory.

Application Data Folder

Returns the full path to the directory containing application data for all users.

Syntax

```
$AppDataDir
```

Examples

`$AppDataDir` -> "C:\ProgramData\Labelstar Office"

Application Folder

Returns the full path to the current application directory.

Syntax

```
$AppDir
```

Examples

`$AppDir` -> "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office"

See also

- Application path
- Installation folder

Application Path

Returns the full path to the current application file.

Syntax

```
$AppPath
```

Examples

$AppPath -> "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office\LabelDesigner.exe"

See also

- Application folder
- Installation folder

Folder Name

Returns the directory information for the specified path string.

Syntax

```
$Dir (path)
```

Parameters

path
Path string

Remarks

Returns the directory information for the specified path or an empty string (""), if the path does not contain directory information.

Examples

Application path = "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office\LabelDesigner.exe"

```
$Dir ("C:\Labels\Label1.lbex") -> "C:\Labels"
$Dir ($AppPath) -> "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office"
```

See also

- File name
- File extension

File Extension

Returns the file extension of the specified path string.

Syntax

```
$Ext (path)
```

Parameter

path
Pfadzeichenfolge

Hinweis

Gibt die Dateierweiterung des angegebenen Pfades (einschließlich ".") oder einen leeren Text (""), wenn der angegebene Pfad keine Erweiterung enthält, zurück.

Beispiele

Programmpfad = "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office\LabelDesigner.exe"

```
$Ext ("C:\Label.lbex") -> ".lbex"
$Ext ($AppPath) -> ".exe"
$Ext ("C:\Label") -> ""
```

See also

- File name
- Folder name

File Name

Returns the file name of the specified path string.

Syntax

```
$FileName (path, [Extension=with_extension])
```

Parameters

path
Path string

with_extension (optional, default = 1)
Specifies whether the file name should be returned with or without the file extension.
0: Return file name without extension
1: Return file name with extension

Examples

Application path = "C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office\LabelDesigner.exe"
Label path = "C:\Labels\Label1.lbex"

```
$FileName ("C:\Labels\Label1.lbex") -> "Label1.lbex"
$FileName ($AppPath) -> "LabelDesigner.exe"
$FileName ($AppPath, Extension=0) -> "LabelDesigner"
$FileName ($LabelPath) -> "Label1.lbex"
$FileName ($LabelPath, Extension=0) -> "Label1"
```

See also

- Folder name
- File extension

Image Folder

Returns the full path to the default image directory.

Syntax

```
$ImageDir
```

See also

- Change default images location
- Label folder

Installation Folder

Returns the full path to the directory in which **Labelstar Office** is installed.

Syntax

```
$InstallDir
```

Remarks

Default installation directory: *C:\Programs\Carl Valentin GmbH\Labelstar Office* or *C:\Programs (x86)\Carl Valentin GmbH\Labelstar Office* on 64-bit computers.

See also

- Application folder
- Application path

Label Folder

Returns the full path to the default label directory.

Syntax

```
$LabelDir
```

See also

- Change default labels location
- Image folder

Math Variables

With these variables it is possible to carry out mathematical calculations directly on the label.

List of Available Mathematical Variables

	Variable	Description
Absolute Value	$Abs	Returns the absolute value of a number.
Calculate average	$Average	Returns the average of two (or more) numbers.
Calculate Math Expression	$Calculate	Returns the result of a mathematical expression.
Max	$Max	Returns the largest value of two (or more) numbers.
Min	$Min	Returns the smallest value of two (or more) numbers.
Calculate Product	$Product	Returns the product of two (or more) numbers.
Calculate Sum	$Sum	Returns the sum of two (or more) numbers.

Absolute Value

Returns the absolute value of a number.

Syntax

```
$Abs (number)
```

Parameters

number
Numeric expression

Remarks

The absolute value of a number is its numeric value without its sign. For example, the absolute value of both 12 and -12 is 12.

Examples

```
$Abs (12) -> "12"
$Abs (-12) -> "12"
$Abs (-12.25) -> "12,25"
$FormatNumber ($Abs (-144), "00000") -> "00144"
```

Calculate Average

Returns the average of two (or more) numbers.

Syntax

```
$Average (number1, number2, ...)
```

Parameters

number
Numeric expression

Examples

```
$Average (10, 2, 20) -> "10,66667"
$Average (10, 2, 21) -> "11"
$Average (10, 20, 30) -> "20"
```

See also

- Calculate math expression
- Calculate product
- Calculate sum

Calculate Math Expression

Returns the result of a mathematical expression.

Syntax

```
$Calculate (expression, [output_format])
```

Parameters

expression
Mathematical expression. For more information, see Mathematical Operators.

output_format (optional)
Specifies how the result should be formatted. For a detailed list of the valid format string, see Standard Numeric Format Strings or Custom Numeric Format Strings.

Examples

ID01 = "-10"
ID02 = "12"

```
$Calculate ((12 * 12) + 20) -> "164"
$Calculate ((12 * 12) + 20, "N2") -> "164,00"
$Calculate (12.5 * 12.5, "0.00") -> "156,25"
$Calculate (<<ID01>> + <<ID02>>) -> "2"
$Calculate ($Abs (<<ID01>>) + <<ID02>>) -> "22"

$Calculate ((12 * 12) / 10) -> "14,4"
$FormatNumber ($Calculate((12 * 12) / 10), "0.00") -> "14,40"
$FormatNumber ($Calculate((12 * 12) / 10), "0") -> "14"

$Calculate (12.4 * 12.0) -> "148,8"
$FormatNumber ($Calculate (12.4 * 12.0), "N2") -> "148,80"
$FormatNumber ($Calculate (12.4 * 12.0), "0") -> "149"
```

ID	Name	Capital	Area	Population	NativeName	Flag
12	United Kingdom	London	244820	60440000	United Kingdom	

```
$Calculate ($DbField ("Europe", "Population") * 2.00) -> 120880000
```

See also

- If-Then-Else-Statement

Mathematical Operators

An operator is a term of a symbol to which one or several expressions and/or operands are handed over as input and which returns a value.

Unary Operators (operators with one operand)

Expression	Description
+x	Identity
-x	Negation
!x	Logical negation

Arithmetic Operators

Expression	Description
x + y	Addition, string concatenation
x - y	Subtraction
x * Y	Multiplication
x / y	Division
x % y	Modulus (calculates the remainder of the two operands)
x ^ y	Power (calculates the y-th power of x)

Compare Operators

Expression	Description
x = y	Equal to
x == y	Equal to (ignore case)
x != y	Not equal to
x <> y	Not equal to (ignore case)
x < y	Less than
x <= y	Less than or equal
x > y	Greater than
x >= y	Greater than or equal

Logical Operators

Expression	Description
x && y	Conditioned And (y is evaluated only if x is true)
x || y	Conditioned Or (y is evaluated only if x is false)

See also

- Calculate math expression
- If-Then-Else-Statement

Max

Returns the largest value of two (or more) numbers.

Syntax

```
$Max (number1, number2, ...)
```

Parameters

number
Numeric expression

Examples

```
$Max (20, 10, 30) -> "30"
$Max (10, 5) -> "10"
$Max (12.25, 17.53, 4.27, 8.93) -> "17,53"
```

See also

- Min

Min

Returns the smallest value of two (or more) numbers.

Syntax

```
$Min (number1, number2, ...)
```

Parameters

number
Numeric expression

Examples

```
$Min (20, 10, 30) -> "10"
$Min (10, 5) -> "5"
$Min (12.25, 17.53, 4.27, 8.93) -> "4,27"
```

Siehe auch

- Max

Calculate Product

Returns the product of two (or more) numbers.

Syntax

```
$Product (number1, number2, ...)
```

Parameters

number
Numeric expression

Examples

```
$Product (10, 20, 30) -> "6000"
$Product (10, 2) -> "20"

$Product (12.25, 17.53, 4.27, 8.93) -> "8188,368"
$FormatNumber ($Product (12.25, 17.53, 4.27, 8.93), N2) -> "8188,37"

ID01 = "20"
ID02 = "15"
ID03 = "2"

$FormatNumber ($Product (<<ID01>>, <<ID02>>, <<ID03>>), "00000") -> "00600"
```

See also

- Calculate math expression
- Calculate average
- Calculate sum

Calculate Sum

Returns the sum of two (or more) numbers.

Syntax

```
$Sum (number1, number2, ...)
```

Parameters

number
Numeric expression

Examples

```
$Sum (10, 20, 30) -> "60"
$Sum (10, 2) -> "12"
$Sum (12.25, 17.53, 4.27, 8.93) -> "42,98"

ID01 = "20"
ID02 = "15"
ID03 = "2"

$FormatNumber ($Sum (<<ID01>>, <<ID02>>, <<ID03>>), "00000") -> "00037"
```

See also

- Calculate math expression
- Calculate average
- Calculate product

Misc Variables

List of Available Variables

	Variable	Description
Check Digit	$CheckDigit	Returns the check digit calculated using the specified method.
Computer Name	$ComputerName	Returns the full computer name of where **Labelstar Office** is running.
Number of Copies	$Copies	Returns the number of copies.
Custom Check Digit	$CustomCheckDigit	Returns the user-defined check digit.
If-Then-Else-Statement	$If	Tests a user-defined condition and returns one result if the condition is true, and another result if the condition is false.
Label Name	$LabelName	Returns the name of the current label file.
Label Number	$LabelNumber	Returns the current label number within a print job.
Label Path	$LabelPath	Returns the complete path name of the current label file.
Label Size	$LabelSize	Returns the label size of the current label file.
Label Type	$LabelType	Returns the label type of the current label file.
Page Name	$PageName	Returns the current page name.
Page Number	$PageNumber	Returns the current page number within a print job.
Printer Name	$PrinterName	Returns the current printer name.
Shift Variable	$Shift	Returns the shift description depending on the current system or printer time.
User Domain Name	$UserDomainName	Returns the network domain name associated with the current user.
User Name	$UserName	Returns the current user name or the domain user name.

Check Digit

Returns the check digit calculated using the specified method.

Syntax

```
$CheckDigit (data, check_digit_method, [AppendTo=append_to])
```

Parameters

data
Data for which the check digit is to be calculated.

check_digit_method
Method after according to which the check digit is to be calculated.

Method	Description
Mod10	Modulo 10 (EAN-13, GTIN-13)
Mod10Luhn	Modulo 10 (Luhn Alogrithm)
Mod11PZN	Modulo 11 (PZN-8)
Mod11UPU	Modulo 11 (UPU S10 - Universal Postal Union S10 Standard)
Mod43	Modulo 43 (Code 39 und LOGMARS Barcodes)
Mod103	Modulo 103 (Code 128)

For a complete list of the supported calculation methods, see here.

append_to (optional)
Specifies where the calculated check digit is to be appended to the data.
Right: Check digit is appended at the end of the data.
Left: Check digit is appended at the beginning of the data.

Examples

NVE = "34012345123456789"

```
$CheckDigit ("12345", Mod10) -> "7"
$CheckDigit (<<NVE>>, Mod10) -> "5"
$CheckDigit (<<NVE>>, Mod10, AppendTo=Right) -> "340123451234567895"
```

See also

- Custom Check Digit
- Check Digit (Printer)

Predefined Check Digit Calculation Methods

Labelstar Office supports a variety of predefined check digit calculation methods. These methods can be used in the Check Digit (System) variable.

- **Code11_1Digit** - Code 11 check digit (1 digit)
- **Code11_2Digits** - Code 11 check digit (2 digits)
- **Code128** - Code 128 internal check digit (Modulo-103)
- **DPIdentcode** - Deutsche Post Identcode check digit. An example of a check digit calculation using this method can be found here.
- **DPLeitcode** - Deutsche Post Leitcode check digit. An example of a check digit calculation using this method can be found here.
- **Ean8** - EAN-8, GTIN-8 check digit
- **Ean13** - EAN-13, GTIN-13 check digit. An example of a check digit calculation using this method can be found here.
- **Ean14** - EAN-14 check digit
- **Ean128** - GS1-128 internal check digit (Modulo-103)
- **Mod10** - Modulo-10 check digit (EAN-13, GTIN-13). An example of a check digit calculation using this method can be found here.
- **Mod10Luhn** - Modulo-10 check digit (Luhn Algorithm). An example of a check digit calculation using this method can be found here.
- **Mod11** - PZN-8 check digit. An example of a check digit calculation using this method can be found here.
- **Mod11UPU** - Modulo-11 check digit (UPU S10 - Universal Postal Union S10 Standard). An example of a check digit calculation using this method can be found here.
- **Mod43** - Modulo-43 check digit (1-digit)
- **Mod103** - Modulo-103 check digit (Code 128)
- **Modulo10** - Code 2 of 5 Interleaved check digit. An example of a check digit calculation using this method can be found here.
- **Modulo10IMPackage** - USPS Intelligent Mail® Barcode check digit
- **Modulo10ItalianPostal25** - Italian Postal 2 of 5 check digit
- **Modulo10KoreaPA** - Korean Postal Authority check digit
- **Modulo10Luhn** - Modulo-10 check digit (Luhn Alogrithm). An example of a check digit calculation using this method can be found here.
- **Modulo10LuhnRev** - Modulo-10 check digit (Luhn Reverse Algorithm)
- **Modulo10Planet** - USPS PLANET check digit
- **Modulo11Pzn** - PZN-8 check digit. An example of a check digit calculation using this method can be found here.
- **Modulo11UPU** - Modulo-11 check digit (UPU S10 - Universal Postal Union S10 Standard). An example of a check digit calculation using this method can be found here.
- **Modulo11Weight7** - Modulo-11 check digit (weight 1 to 7)
- **Modulo11Weight9** - Modulo-11 check digit (weight 2 to 9)
- **Modulo11Weight10** - Modulo-11 check digit (weight 2 to 10)
- **Modulo16** - Codabar check digit
- **Modulo23PPSN** - PPSN check digit (Personal Public Service Number)
- **Modulo36** - DPD (ISO/IES 7064) check digit
- **Modulo37PotentialWeight2** - Modulo-37 check digit (ISO 7064, weight 2 to 2^13)
- **Modulo43** - Modulo-43 check digit (1 digit)
- **Module47_2Digits** - Modulo-47 check digit (2 digits)
- **Msi1Digit** - MSI check digit (1 digit)
- **Msi2Digits** - MSI check digit (2 digits)
- **Plessey** - Plessey check digit

- **Postnet** - USPS PostNet check digit
- **RoyalMail4State** - Royal Mail 4-State check digit
- **UpcA** - UPC-A, GTIN-12 check digit
- **UpcE** - UPC-E check digit
- **Vin** - VIN check digit (Vehicle Identification Number)

Computer Name

Returns the full computer name of where **Labelstar Office** is running.

Syntax

```
$ComputerName
```

See also

- User Domain Name
- User Name

Number of Copies

Returns the number of copies.

Syntax

```
$Copies
```

Examples

Number of copies = 80
Label number = 5

$FormatNumber ($LabelNumber, "000") - $FormatNumber ($Copies, "000") -> "005 - 080"

Custom Check Digit

Returns the user-defined check digit.

Note

Labelstar Office supports a variety of predefined check digit calculation methods. If none of these methods fit, a user-defined check digit can also be calculated based on modulo-10 algorithm used to calculate check digits for EAN-13/GTIN-13 barcodes.

Syntax

```
$CustomCheckDigit (data)
```

Parameters

data
Data for which the check digit is to be calculated.

Internal Parameters

Modulo
Modulo factor

Append to
Specifies where the calculated check digit is to be appended to the data.
None: Return check digit only
Right: Check digit is appended at the end of the data.
Left: Check digit is appended at the beginning of the data.

Weights
List of comma-separated values or range of values.
Example: "1,3,6-9"

Apply weights
Indicates whether the weights are applied **Left-to-right** or **Right-to-left** to the data.

Product processing
Indicates how the sum of weight x digit should be calculated.

Subtract result from
If this option is activated, the result of the check digit calculation will be subtracted from this value.

Examples

Calculation of a user-defined check digit analogous to Modulo 10 (EAN)

Modulo: 10
Weights: 3,1
Apply weights: Right-to-left
Product procession: Total of products
Subtract result from: 10

Digits	3 4 0 1 2 3 4 5 1 2 3 4 5 6 7 8
Weights	1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3
Total of products	3 + 12 + 0 + 3 + 2 + 9 + 4 + 15 + 1 + 6 + 3 + 12 + 5 + 18 + 7 + 24 = 124
Remainder mod 10	124 mod 10 = 4 (124/10 = 12 remainder 4)
Subtract from 10	10 - 4 = 6
Check digit	**6**

Calculation of a user-defined check digit analogous to Modulo 10 (Luhn Algorithm)

Modulo: 10
Weights: 2,1
Apply weights: Right-to-left
Product processing: Total of product digits
Subtract result from: 10

Digits	3 4 0 1 2 3 4 5 1 2 3 4 5 6 7 8
Weights	1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
Total of product digits	3 + 8 + 0 + 2 + 2 + 6 + 4 + 1 + 1 + 4 + 3 + 8 + 5 + 3 + 7 + 7 = 64
Remainder mod 10	64 mod 10 = 4 (64/10 = 6 remainder 4)
Subtract from 10	10 - 4 = 6
Check digit	**6**

Calculation of a user-defined check digit analogous to Modulo 11 (PZN-8)

Modulo: 11
Weights: 1-7
Apply weights: Left-to-right
Product processing: Total of products

Digits	3 6 3 1 9 4 2
Weights	1 2 3 4 5 6 7
Total of products	3 + 12 + 9 + 4 + 45 + 24 + 14 = 111
Remainder mod 11	111 mod 1 = 1 (111/11 = 10 remainder 1)
Check digit	**1**

See also

- Check Digit (System)
- Check Digit (Printer)

If-Then-Else-Statement

Tests a user-defined condition and returns one result if the condition is true, and another result if the condition is false.

Syntax

```
$If (condition, value_if_true, value_if_false)
```

Parameters

condition
The condition to be evaluated. For more information, see Mathematical Operators.

value_if_true
The value that is returned if condition evaluates to **TRUE (1)**.

value_if_false
The value that is returned if condition evaluates to **FALSE (0)**.

Examples

	ID	Name	Capital	Area	Population	NativeName	Flag
▶	12	United Kingdom	London	244820	60440000	United Kingdom	

```
$If ($DbField ("Europe", "Area") <= 250000, "*", "**") -> "*"
```

Check whether a database field is empty

```
$If ($Length ($DBField (...)) = 0, "The database field is empty.", "The database field is not
empty.")
$If ($IsEmpty ($DBField (...)), "The database field is empty.", "The database field is not empty.")
```

Nested Queries

```
$If ($IsEqual (<<AirportCode>>, "LAX"), "Los Angeles", $If ($IsEqual (<<AirportCode>>, "DEN"),
"Denver", $If ($IsEqual (<<AirportCode>>, "JFK"), "John F. Kennedy", $If ($IsEqual (<<AirportCode>>,
"SEA"), "Seattle", "Other")))) International Airport
```

or

```
$If (<<AirportCode>> = "LAX", "Los Angeles", $If (<<AirportCode>> = "DEN", "Denver", $If
(<<AirportCode>> = "JFK", "John F. Kennedy", $If (<<AirportCode>> = "SEA", "Seattle", "Other"))))
International Airport
```

AirportCode = "LAX" -> "Los Angeles International Airport"
AirportCode = "DEN" -> "Denver International Airport"
AirportCode = "JFK" -> "John F. Kennedy International Airport"
AirportCode = "SEA" -> "Seattle International Airport"
AirportCode = "LAS" -> "Other International Airport"
AirportCode = "DFW" -> "Other International Airport"

See also

- Calculate math expression

Label Name

Returns the name of the current label file.

Syntax

```
$LabelName
```

See also

- Label Path
- Label Size
- Label Type

Label Number

Returns the current label number within a print job.

Syntax

```
$LabelNumber
```

Examples

Number of copies = 80
Label number = 5

$FormatNumber ($LabelNumber, "000") - $FormatNumber ($Copies, "000") -> "005 - 080"

See also

- Page Name
- Page Number

Label Path

Returns the complete path name of the current label file.

Syntax

```
$LabelPath
```

See also

- Label Name
- Label Size
- Label Type

Label Size

Returns the label size of the current label file.

Syntax

```
$LabelSize
```

See also

- Label Name
- Label Path
- Label Type

Label Type

Returns the label type of the current label file.

Syntax

```
$LabelType
```

See also

- Label Name
- Label Path
- Label Size

Page Name

Returns the current page name.

Syntax

```
$PageName
```

See also

- Label Number
- Page Number

Page Number

Returns the current page number within a print job.

Syntax

```
$PageNumber
```

See also

- Label Number
- Page Name

Printer Name

Returns the current printer name.

Syntax

```
$PrinterName
```

Shift Variable

Returns the shift description depending on the current system or printer time.

Syntax

```
$Shift
```

Examples

Early shift: 6:00 - 13:59
Late shift: 14:00 - 21:59
Night shift: 22:00 - 5:59

System Variable (TrueType Font)

`$Shift` -> "Early shift" (8:20)
`$Shift` -> "Late shift" (15:30)

Printer Variable (Printer Font)

`$Shift` -> "=SH()"

See also

- Define shift times

Define Shift Times

To define the shift times, please proceed as follows:

1. Select **Label Properties** and click **Shifts** in the **Settings** group.
   The **Shifts** dialog box opens.
2. Do the following:
   - **Add new shift** Click **Add** to insert a new item to the list. Enter a unique **Shift Name** (this will be printed on the lable) and the **Start** and **End** of the shift.
     Note: It is important that the shift times do not overlap. If a shift starts at 12:00, the previous shift must end at 11:59 at the latest.

	Wrong	Right
Early shift	6:00-14:00	6:00-13:59
Late shift	14:00 - 22:00	14:00 - 21:59
Night shift	22:00 - 6:00	22:00 - 5:59

   - **Edit shift** Select an item and click .
   - **Delete shift** Select an item and click .
   - **Move shift** Select an item and click **Move up** or **Move down** to move the item up or down one position, or drag the selected item to the desired position within the list.
3. Click **OK** to save settings.

See also

- Shift Variable

User Domain Name

Returns the network domain name associated with the current user.

Syntax

```
$UserDomainName
```

See also

- Computer Name
- User Name

User Name

Returns the current user name or the domain user name.

Syntax

```
$UserName
```

See also

- Computer Name
- User Domain Name

Format Strings

General Definitions

- Formatting is the process of converting a value to a string representation.
- Format specifiers are special characters that determine how to format the output based on which specifier was used. Format specifiers are also known as format strings. For example, 'c' and 'C' are format specifiers for converting a number to a string that represent a currency amount.
- Parsing is the process of creating a value from a string represantation. Parsing is the opposite of formatting.

Labelstar Office provides three types of format strings:

- **Numeric Format Strings**

  Used to create strings from a number. For more information, see Standard Numeric Format Strings and Custom Numeric Format Strings.
- **Date and Time Format Strings**

  Used to create strings from a date and time. For more information, see Standard Date and Time Format Strings and Custom Date and Time Format Strings.
- **Text Format Strings**

  Used to create masked strings. For more information, see Text Format Strings.

Standard Numeric Format Strings

Standard numeric format strings are used to format common numeric types. A standard numeric format string takes the form Axx, where A is an alphabetic character called the format specifier, and xx is an optional integer called the precision specifier. The precision specifier ranges from 0 to 99 and affects the number of digits in the result. Any numeric format string that contains more than one alphabetic character, including white space, is interpreted as a custom numeric format string. For more information, see Custom Numeric Format Strings.

The following table describes the standard numeric format specifiers.

Format Specifier	Name	Description	Examples
C oder c	Currency	Result: A currency value. Precision specifier: Number of decimal digits.	123.456 ("C", en-US) -> $123.46 123.456 ("C", fr-FR) -> 123,46 € 123.456 ("C", ja-JP) -> ¥123 -123.456 ("C3", en-US) -> ($123.456) -123.456 ("C3", fr-FR) -> -123,456 € -123.456 ("C3", ja-JP) -> -¥123.456
D oder d	Decimal	Result: Integer digits with optional negative sign. Precision specifier: Minimum number of digits.	1234 ("D") -> 1234 -1234 ("D6") -> -001234
E oder e	Exponential (scientific)	Result: Exponential notation. Precision specifier: Number of decimal digits.	1052.0329112756 ("E", en-US) -> 1.052033E+003 1052.0329112756 ("e", fr-FR) -> 1,052033e+003 -1052.0329112756 ("e2", en-US) -> -1.05e+003 -1052.0329112756 ("E2", fr_FR) -> -1,05E+003
F oder f	Fixed-point	Result: Integral and decimal digits with optional negative sign. Precision specifier: Number of decimal digits.	1234.567 ("F", en-US) -> 1234.57 1234.567 ("F", de-DE) -> 1234,57 1234 ("F1", en-US) -> 1234.0 1234 ("F1", de-DE) -> 1234,0 -1234.56 ("F4", en-US) -> -1234.5600 -1234.56 ("F4", de-DE) -> -1234,5000
G oder g	General	Result: The most compact of either fixed-point or scientific notation. Precision specifier: Number of significant digits.	-123.456 ("G", en-US) -> -123.456 123.456 ("G", sv-SE) -> -123,456 123.4546 ("G4", en-US) -> 123.5 123.4546 ("G4", sv-SE) -> 123,5 -1.234567890e-25 ("G", en-US) -> -1.23456789E-25 -1.234567890e-25 ("G", sv-SE) -> -1,23456789E-25
N oder n	Number	Result: Integral and decimal digits, group separators, and a decimal separator with optional negative sign. Precision specifier: Desired number of decimal places.	1234.567 ("N", en-US) -> 1,234.57 1234.567 ("N", ru-RU) -> 1 234,57 1234 ("N1", en-US) -> 1,234.0 1234 ("N1", ru-RU) -> 1 234,0 -1234.56 ("N3", en-US) -> -1,234.560 -1234.56 ("N3", ru-RU) -> -1 234,560
P oder p	Percent	Result: Number multiplied by 100 and displayed with a percent symbol. Precision specifier: Desired number of decimal places.	1 ("P", en-US) -> 100.00 % 1 ("P", fr-FR) -> 100,00 % -0.39678 ("P1", en-US) -> -39.7 % -0.39678 ("P1", fr-FR) -> -39,7 %
R oder r	Round-trip	Result: A string that can round-trip to an identical number. Precision specifier: Ignored.	123456789.12345678 ("R") -> 123456789.12345678 -1234567890.12345678 ("R") -> -1234567890.1234567
X oder x	Hexadecimal	Result: A hexadecimal string.	255 ("X") -> FF

		Precision specifier: Number of digits in the result string.	-1 ("x") -> ff 255 ("x4") -> 00ff -1 ("X4") -> 00FF

Custom Numeric Format Strings

You can create a custom numeric format string, which consists of one or more custom numeric specifiers, to define how to format numeric data. A custom numeric format string is any format string that is not a Standard Numeric Format Strings.

The following table describes the custom numeric format specifiers.

Format Specifier	Name	Description	Examples
0	Zero placeholder	Replaces the zero with the corresponding digit if one is present; otherwise, zero appears in the result string.	1234.5678 ("00000") -> 01235 0.45678 ("0.00", en-US) -> 0.46 0.45678 ("0.00", fr-FR) -> 0,46
#	Digit placeholder	Replaces the pound sign with the corresponding digit if one is present; otherwise, no digit appears in the result string.	1234.5678 ("#####") -> 1235 0.45678 ("#.##", en-US) -> .46 0.45678 ("#.##", fr-FR) -> ,46
.	Decimal point	Determines the location of the decimal separator in the result string.	0.45678 ("0.00", en-US) -> 0.46 0.45678 ("0.00", fr-FR) -> 0,46
,	Group separator and number scaling	Serves as both a group separator and a number scaling specifier. As a group separator, it inserts a localized group separator character between each group. As a number scaling specifier, it divides a number by 1000 for each comma specified.	Group separator specifier: 2147483647 ("##,#", en-US) -> 2,147,483,647 2147483647 ("##,#", es-ES) -> 2.147.483.647 Scaling specifier: 2147483647 ("#,#,,", en-US) -> 2,147 2147483647 ("#,#,,", es-ES) -> 2.147
%	Percent placeholder	Multiplies a number by 100 and inserts a localized percentage symbol in the result string.	0.3697 ("%#0.00", en-US) -> %36.97 0.3697 ("%#0.00", el-GR) -> %36,97 0.3697 ("##.0 %", en-US) -> 37.0 % 0.3697 ("##.0 %", el-GR) -> 37,0 %
‰	Per mille placeholder	Multiplies a number by 1000 and inserts a localized per mille symbol in the result string.	0.03697 ("#0.00‰", en-US) -> 36.97‰ 0.03697 ("#0.00‰", ru-RU) -> 36,97‰
\	Escape character	Causes the next character to be interpreted as a literal rather than as a custom format specifier.	987654 ("\###00\#") -> #987654#
;	Section separator	Defines sections with separate format strings for positive, negative, and zero numbers.	15 (#;minus #;Null) -> 15 -15 (#;minus #;Null) -> minus 15 0 (#;minus #;Null) -> Null
Any other character		The character is copied to the result string unchanged.	68 ("# °") -> 68 °

Standard Date and Time Format Strings

A standard date and time format string uses a single format specifier to define the text representation of a date and time value. Any date and time format string that contains more than one character, including white space, is interpreted as a custom date and time format strin. For more information, see Custom Date and Time Format Strings.

The following table describes the standard date and time format specifiers.

Format Specifier	Description	Eamples
d	Short date pattern	6/15/2009 13:45:30 -> 15.06.2009 (de-DE) 6/15/2009 13:45:30 -> 6/15/2009 (en-US) 6/15/2009 13:45:30 -> 15/06/2009 (fr-FR) 6/15/2009 13:45:30 -> 2009/06/15 (ja-JP)
D	Long date pattern	6/15/2009 13:45:30 -> Montag, 15. Juni 2009 (de-DE) 6/15/2009 13:45:30 -> Monday, June 15, 2009 (en-US) 6/15/2009 13:45:30 -> 15 июня 2009 г.(ru-RU) 6/15/2009 13:45:30 -> Montag, 15.Juni 2009 (de-DE)
f	Full date/time pattern (short time)	6/15/2009 13:45:30 -> Montag, 15. Juni 2009 13:45 (de-DE) 6/15/2009 13:45:30 -> Monday, June 15, 2009 1:45 PM (en-US) 6/15/2009 13:45:30 -> Höhle 15 juni 2009 13:45 (sv-SE) 6/15/2009 13:45:30 -> Δευτέρα, 15 Ιουνίου 2009 1:45 μμ (el-GR)
F	Full date/time pattern (long time)	6/15/2009 13:45:30 -> Montag, 15. Juni 2009 13:45:30 (de-DE) 6/15/2009 13:45:30 -> Monday, June 15, 2009 1:45:30 PM (en-US) 6/15/2009 13:45:30 -> den 15 juni 2009 13:45:30 (sv-SE) 6/15/2009 13:45:30 -> Δευτέρα, 15 Ιουνίου 2009 1:45:30 μμ (el-GR)
g	General date/time pattern (short time)	6/15/2009 13:45:30 -> 15.06.2009 13:45 (de-DE) 6/15/2009 13:45:30 -> 6/15/2009 1:45 PM (en-US) 6/15/2009 13:45:30 -> 15/06/2009 13:45 (es-ES) 6/15/2009 13:45:30 -> 2009/6/15 13:45 (zh-CN)
G	General date/time pattern (long time)	6/15/2009 13:45:30 -> 15.06.2009 13:45:30 (de-DE) 6/15/2009 13:45:30 -> 6/15/2009 1:45:30 PM (en-US) 6/15/2009 13:45:30 -> 15/06/2009 13:45:30 (es-ES) 6/15/2009 13:45:30 -> 2009/6/15 13:45:30 (zh-CN)
l	Short date pattern (last day of month)	6/15/2009 13:45:30 -> 30.06.2009 (de-DE) 6/15/2009 13:45:30 -> 6/30/2009 (en-US) 6/15/2009 13:45:30 -> 30/06/2009 (fr-FR) 6/15/2009 13:45:30 -> 2009/06/30 (ja-JP)
L	Long date pattern (last day of month)	6/15/2009 13:45:30 -> Dienstag, 30. Juni 2009 (de-DE) 6/15/2009 13:45:30 -> Tuesday, June 30, 2009 (en-US) 6/15/2009 13:45:30 -> 30 июня 2009 г. (ru-RU) 6/15/2009 13:45:30 -> Dienstag, 30. Juni 2009 (de-DE)
M oder m	Day/Month pattern	6/15/2009 13:45:30 -> 15. Juni (de-DE) 6/15/2009 13:45:30 -> June 15 (en-US) 6/15/2009 13:45:30 -> 15juni (da-DK) 6/15/2009 13:45:30 -> 15 Juni (id-ID)
R oder r	RFC1123 pattern	6/15/2009 13:45:30 - > Mon, 15 Jun 2009 13:45:30 GMT
s	Sortable date/time pattern	6/15/2009 13:45:30 -> 2009-06-15T13:45:30
t	Short time	6/15/2009 13:45:30 -> 13:45 (de-DE) 6/15/2009 13:45:30 -> 1:45 PM (en-US) 6/15/2009 13:45:30 -> 13:45 (hr-HR)

		6/15/2009 13:45:30 -> 01:45 م (ar-EG)
T	Long time	6/15/2009 13:45:30 -> 13:45:30 (de-DE) 6/15/2009 13:45:30 -> 1:45:30 PM (en-US) 6/15/2009 13:45:30 -> 13:45:30 (hr-HR) 6/15/2009 13:45:30 -> 01:45:30 م (ar-EG)
u	Universal sortable date/time pattern	6/15/2009 13:45:30 -> 2009-06-15 13:45:30Z
U	Universal full date/time pattern	6/15/2009 13:45:30 -> Montag, 15. Juni 2009 11:45:30 (de-DE) 6/15/2009 13:45:30 -> Monday, June 15, 2009 8:45:30 PM (en-US) 6/15/2009 13:45:30 -> den 15 juni 2009 20:45:30 (sv-SE) 6/15/2009 13:45:30 -> Δευτέρα, 15 Ιουνίου 2009 8:45:30 μμ (el-GR)
Y oder y	Year/Month pattern	6/15/2009 13:45:30 -> Juni 2009 (de-DE) 6/15/2009 13:45:30 -> June, 2009 (en-US) 6/15/2009 13:45:30 -> juni 2009 (da-DK) 6/15/2009 13:45:30 -> Juni 2009 (id-ID)

Custom Date and Time Format Strings

A custom format string consists of one or more custom date and time format specifiers. Any string that is not a standard date and time format string is interpreted as a custom date and time format string.

Custom Format Specifiers for Date

Format Specifier	Description	Range	Examples
d	Day of month	1-31	6/1/2009 13:45:30 -> 1 6/15/2009 13:45:30 -> 15
dd	Two-digit day of month Single-digit day values are preceded by 0.	01-31	6/1/2009 13:45:30 -> 01 6/15/2009 13:45:30 -> 15
ddd	Abbreviated weekday name		6/15/2009 13:45:30 -> Mo (de-DE) 6/15/2009 13:45:30 -> Mon (en-US) 6/15/2009 13:45:30 -> Пн (ru-RU) 6/15/2009 13:45:30 -> lun. (fr-FR)
dddd	Full weekday name		6/15/2009 13:45:30 -> Montag (de-DE) 6/15/2009 13:45:30 -> Monday (en-US) 6/15/2009 13:45:30 -> понедельник (ru-RU) 6/15/2009 13:45:30 -> lundi (fr-FR)
dm	Last day of month		6/15/2009 01:09:30 -> 30 2/3/2016 01:09:30 -> 29 (Schaltjahr)
dwx	Day of week For x any ASCII character can be used, from this is counted consecutively started with Sunday.		6/15/2009 13:45:30 -> 1 (dw0) 6/15/2009 13:45:30 -> B (dwA)
j	Day of Year	1-366	1/1/2010 13:24:20 -> 1 3/19/2010 12:52:10 -> 78 12/31/2010 15:20:15 -> 365 12/31/2012 15:20:15 -> 366 (Schaltjahr)
jjj	Three-digit day of year Shorter values are preceded by 0.	001-336	1/1/2010 13:24:20 -> 001 3/19/2010 12:52:10 -> 078 12/31/2010 15:20:15 -> 365 12/31/2012 15:20:15 -> 366 (Schaltjahr)
M	Month	1-12	6/15/2009 13:45:30 -> 6
MM	Two-digit month Single-digit values are preceded by 0.	01-12	6/15/2009 13:45:30 -> 06
MMM	Abbreviated month name		6/15/2009 13:45:30 -> Jun (de-DE) 6/15/2009 13:45:30 -> Jun (en-US) 6/15/2009 13:45:30 -> jun (fr-FR) 6/15/2009 13:45:30 -> 6月 (zh-CN)
MMMM	Full month name		6/15/2009 13:45:30 -> Juni (de_DE) 6/15/2009 13:45:30 -> June (en-US) 6/15/2009 13:45:30 -> juni (da-DK) 6/15/2009 13:45:30 -> 六月 (zh-CN)
w	Calendar week	1-53	5/18/2016 13:14:30 -> 20 (de-DE) 5/18/2016 13:14:30 -> 21 (en-US)
ww	Two-digit calendar week Single-digit values are preceded by 0.	01-53	
y	Year	0-99	1/1/0001 00:00:00 -> 1 1/1/0900 00:00:00 -> 0 1/1/1900 00:00:00 -> 0

			6/15/2009 13:45:30 -> 9 3/21/2018 00:00:00 -> 18
yy	Two-digit year Single-digit values are preceded by 0.	00-99	1/1/0001 00:00:00 -> 01 1/1/0900 00:00:00 -> 00 1/1/1900 00:00:00 -> 00 6/15/2009 13:45:30 -> 09
yyyy	Four-digit year Single-digit values are preceded by 0.		1/1/0001 00:00:00 -> 0001 1/1/0900 00:00:00 -> 0900 1/1/1900 00:00:00 -> 1900 6/15/2009 13:45:30 -> 2009
Y	Year	0-9	6/15/2009 13:45:30 -> 9 3/21/2018 00:00:00 -> 8
/	Date separator		6/15/2009 13:45:30 -> . (de-DE) 6/15/2009 13:45:30 -> / (en-US) 6/15/2009 13:45:30 -> - (ar-DZ) 6/15/2009 13:45:30 -> . (tr-TR)

Custom Format Specifiers for Time

Format Specifier	Description	Range	Examples
h	Hour (12-hour format)	1-12	6/15/2009 01:45:30 -> 1 6/15/2009 13:45:30 -> 1
hh	Two-digit hour (12-hour format) Single-digit values are preceded by 0.	01-12	6/15/2009 01:45:30 -> 01 6/15/2009 13:45:30 -> 01
H	Hour (24-hour format)	0-23	6/15/2009 01:45:30 -> 1 6/15/2009 13:45:30 -> 13
HH	Two-digit hour (24-hour format) Single-digit values are preceded by 0.	00-23	6/15/2009 01:45:30 -> 01 6/15/2009 13:45:30 -> 13
m	Minutes	0-59	6/15/2009 01:09:30 -> 9 6/15/2009 13:09:30 -> 9
mm	Two-digit minutes Single-digit values are preceded by 0.	00-59	6/15/2009 01:09:30 -> 09 6/15/2009 13:09:30 -> 09
s	Seconds	0-59	6/15/2009 13:45:09 -> 9
ss	Two-digit seconds Single-digit values are preceded by 0.	00-59	6/15/2009 13:45:09 -> 09
:	Time separator		6/15/2009 13:45:30 -> : (de-DE) 6/15/2009 13:45:30 -> : (en-US) 6/15/2009 13:45:30 -> . (it-IT) 6/15/2009 13:45:30 -> : (ja-JP)

Text Format Strings

With the help of the text format string, an output formatting for the text display can be defined.

This makes it possible to display data on the label in a form that is easier for people to read. For example, a long sequence of numbers like the IBAN "DE89123447624758123400" can be read better if it is divided into blocks "DE89 1234 4762 4758 1234 00".

The following table lists the placeholder and literal characters for an output mask:

Character	Description	Examples
0	Digit or 0	abc-123-DEF ("00000") -> "12300" abc-123-DEF ("!00000") -> "00123"
9	Digit or space	abc-123-DEF ("99999") -> "123 " abc-123-DEF ("!99999") -> " 123"
#	Digit (optional)	abc-123-DEF ("#####") -> "123"
L	Letter or space	abc-123-DEF ("LLLLL") -> "abcDE"
?	Letter (optional)	
A	Alphanumeric character or space	abc-123-DEF ("AAAAA") -> abc12
a	Alphanumeric characters (optional)	
&	Any character or space	abc-123-DEF ("&&&&&") -> "abc-1"
C	Any character (optional)	

In addition, the following characters have sepcial meaning:

Character	Description	Examples
<	Converts all following characters (a-Z) in small letters.	abcDEF ("<LLLLLL") -> "abcdef"
>	Converts all following characters (a-Z) in capital letters.	abcDEF (">LLLLLL") -> "ABCDEF"
|	Terminates a previous < or >.	abcDEF (">LL<LL|LL") -> ABcdEF
!	Right-aligns the result string.	123 ("00000") -> "12300" 123 ("!00000") -> "00123"
\	Treats the next character as literal text rather than a mask character.	abcDEF (LLL\LLLL) -> "abcLDEF"

All other characters are treated as literals.

Regular Expressions

With the help of **Regular Expressions**, certain patterns can be searched for within a string of references. There is a syntax attached to the expressions which appears strange upon first glance, but can be used in many different ways.

Simple Search Query

What does a search term have to look like if you want to find the words "car" and "cars" in a text? Just use "car" as the search term and you will also get "motorcars" and "scar". It is better to formulate it as follows: "Search for the word 'car' with and without an 's' at the end." Such formulations are the power of **Regular Expressions**. So for the "car" search, the expression looks like this:

```
Cars?
```

The question mark is a placeholder which checks whether the previous character (in this case, it's the letter 's') appears or not. As a result, the search will find all the "car" and "cars" terms, but also all other variations on "car", as the expression doesn't yet know that it only has to find the whole words themselves. The changed expression looks like this:

```
\bAutos?\b
```

The \b functions as a limitation for the word. With it, the search will find all of the "car" and "cars" terms if both of these terms appear in the text as individual words.

Finding Patterns of Digits

What does a search term have to look like if you want to check whether a text contains digits between 10 and 19? The search term would have to be: "Search for a two-digit number where the first digit is 1".

```
\b1[0-9]\b
```

You already know the \b. They are making sure that only two-digit numbers are found. Otherwise, rows of numbers such as "12345" would also be found. Square brackets contain a group of digits to find alternatives. In order for all digits between 0 and 9 to be found, you would actually have to use [0123456789]. This can be shortened to [0-9].

Multiple square brackets can also be used alongside each other. Let's say you want to check whether a text contains an odd number between 801 and 999. Then the expression will look like this:

```
\b[89][0-9][13579]\b
```

First of all, it checks whether the first number is 8 or 9, then every number between 0 and 9 is permitted and the last number has to be an odd number, e.g. 1, 3, 5, 7 or 9.

Using Quantifiers

To find a number between 1 and 999 in a text, you can use the following expression:

```
\b[0-9]?[0-9]?[0-9]\b
```

You can simplify the expression by using quantifiers.

```
\b[0-9]{1,3}\b
```

The quantifier always refers to the previous character and determines how many successive times the figure can appear. In our example, the expression is asking for a minimum of one digit between 0 and 9 and is allowing a maximum of three digits.

Finding Patterns of Digits and Letters

Of course, the square brackets do not just have to contain numbers; they can also be combined with letters. In order to find all of the 5-figure combinations in a text, the expression looks like this:

```
\b[0-9a-zA-Z]{5}\b
```

This pattern will recognise every 5-figure combination which consists of the numbers 0 to 9 and the letters a-Z. All words which contain other letters (e.g. undescores) are not recognised. Therefore, in order for words which contain underscores to also be recognised, the expression must be changed as follows:

```
\b[0-9a-zA-ZÄäÖöÜü]{5}\b
```

Or Operators

If you want different terms to lead to a match in a text, then these terms need to be combined with OR. For example:

```
\b(Google|Yahoo|MSN)\b
```

This pattern has the option of finding the terms "Google", "Yahoo" or "MSN".

You can find a detailed description of the individual symbols which can be used in **Regular Expressions** here.

Regular Expression Symbols

Symbol	Description	Pattern	Input	Matches
.	Matches any character except line break characters \r and \n.	`a.e`	"nave" "water"	"ave" "ate"
*	Matches the previous element zero or more times.	`\d*\.\d`		".0", "19.9", "219.9"
?	Matches the previous element zero or one time.	`"rai?n"`		"ran", "rain"
+	Matches the previous element one or more times.	`"be+"`	"been" "bent"	"bee" "be"
{n}	Matches the previous element exactly n times.	`",\d{3}"`	"1,043.6" "9,876,543,210"	",043" ",876", ",543", ",210"
{n,}	Matches the previous element at least n times.	`"\d{2,}"`		"166", "29", "1930"
{n,m}	Matches the previous element at least n times, but no more than m times.	`"\d{3,5}"`	"193024"	"19302"
[]	Checks whether the current character matches a character in the square brackets.	`[ae]`	"gray" "lane"	"a" "a", "e"
[^]	Verifies that the current character does not match any character in the square brackets.	`[^aei]`	"reign"	"r", "g", "n"
\d	Matches any digit; identical with [0-9]	`\d`	"4=IV"	"4"
\D	Matches any character except digits; identical with [^0-9]	`\D`	"4=IV"	"=", "I", "V"
\s	Matches any white space character, including spaces, tabulators and line breaks	`\w\s`	"ID A1.3"	"D "
\S	Matches any character except white space characters	`\w\S`	"ID A1.3"	"ID", "AI"
\w	Matches any word character, including letters, digits and underscores; identical with [a-zA-Z0-9_]	`\w`	"ID A1.3"	"I", "D", "A", "1", "3"
\W	Matches any character except word characters; identical with [^a-zA-Z0-9_]	`\W`	"ID A1.3"	" ", "."
^	The match must start at the beginning of the string or line.	`^\d{3}`	"901-333"	"901"
$	The match must occur at the end of the string or line.	`\d{3}$`	"901-333"	"333"
\b	The match must occur on a boundary between a word characters (\w) and a non-word character (\W).	`\b\w+\s\w+\b`	"them theme them them"	"them theme", "them them"
|	Matches any one element separated by the vertical bar (|) character.	`th(e|is|at)`	"This is the day. "	"the", "this"
(?i)	Sets case insensitivity option in the middle of a pattern.	`\b(?i)a(?-i)a\w+\b`	"aardvark AAAuto aaaAuto Adam breakfast"	"aardvark", "aaaAuto"
(?-i)	Disables case insensitivity option in the middle of a pattern.			

Examples for the **Regular Expression** usage, see here.

Printer Variables

With the help of these variables printer-internal variables can be defined on the label. In contrast to system variables, printer-internal variables are managed and calculated during the print order by the printer.

Note

- Printer variables can be used only in text boxes with printer fonts and barcodes which are not graphically transferred.
- For each text or barcode field only **one** printer variable can be defined.

Supported Printer Variables

	Variable	Description
Check Digit	$PrnCheckDigit	Defines a printer-internal check digit.
Counter	$PrnCounter	Defines a printer-internal alphanumeric counter.
Numeric Counter	$PrnCounterExt	Defines a printer-internal numeric counter.
Date/Time	$PrnDateTime	Defines a printer-internal date and time variable.
Database Field	$PrnDbField	Defines a printer-internal database field.
Field Link	$PrnFieldLink	Defines a printer-internal field link.
Substring	$PrnSubstring	Defines a printer-internal substring.
User Input	$PrnUserInput	Defines a printer-internal user input.
SAPscript Variable	$SAPField	Defines a printer-internal SAPscript variable.

Check Digit

Defines a printer-internal check digit.

Syntax

```
$PrnCheckDigit (text_or_field_name, check_digit_method)
```

Parameters

text_or_field_name
Data (field name or text constant) for which the check digit is to be calculated. A text constant must be enclosed in quotes ("). The quotation marks will not be printed.
Note: Only printer-internal fields can be used.

check_digit_method
Method after according to which the check digit is to be calculated.

Method	Description
Mod10	Modulo 10 (EAN-13, GTIN-13)
Mod11	Modulo 11 (PZN8)
Mod43	Modulo 43
Mod47Weight15	Modulo 47 (weighting 15)
Mod47Weight20	Modulo 47 (weighting 20)
Mod103	Modulo 103 (Code 128)

Examples

ID01 = "9876543210987"

```
$PrnCheckDigit (ID01, Mod10) -> =CD(0;0;0) -> "9"
$PrnCheckDigit ("123456789012", Mod10) -> =CD("123456789012";0;0) -> "8"
```

See also

- Check Digit (System)
- Custom Check Digit (System)

Counter

Defines a printer-internal alphanumeric counter.

Syntax

```
$PrnCounter (start_value, [Prompt=prompt_text], [UpdateInterval=update_interval],
[Increment=increment], [StartIndex=start_index], [Radix=radix], [Mode=mode], [ResetTime=reset_time],
[ResetValue=reset_value], [UpdateValue=update_value])
```

Parameters

start_value
Current start value.
Note: The number of digits of the start value determines the output format. A maximum of 18 digits can be displayed.

prompt_text (optional)
If a prompt text is defined, the start value must be entered at print start.

update_interval (optional, default = 1)
Indicates how often the variable is to be updated during a print job.
1: After each label
n: After n labels

increment (optional, default = 1)
Value to increment or decrement the counter.

start_index (optional, default = 0)
Defines the position within the start value at which the counter begins to count.

radix (optional, default = 10)
Radix, base of the counter
1: Alphabetical (A-Z)
2: Binary (0, 1)
8: Octal (0-7)
10: Decimal (0-9)
16: Hexadecimal (0-9, A-F)
36: Alphanumeric (0-9, A-Z)

mode (optional, default = 0)
Operating mode
0: Keep start value
1: Reset start value at print start
2: Enter start value at printer
3: Enter start value (= last end value) at printer
4: Reset start value at cycle end
5: Reset start value by I/O signal
6: Reset start value time-controlled
7: Reset start value time-controlled (enter start value at printer)

reset_time (optional, only relevant for operating mode 6 and 7)
Time to which the start value is to be reset.
Format: "HH:MM"

reset_value (optional, only relevant for operating mode 6 and 7)
Value to which the start value is to be reset. If no value is indicated, the counter is reset to its original start value.

update_value (optional, only relevant for operation mode 0, 1, 4, 5, and 6, default = 0)
Specifies whether the start value on the screen should be overwritten or not.
0: Keep start value
1: Overwrite start value

Examples

`$PrnCounter ("0001", Mode=1)` -> `=CN(10;1;4;+1;1)0001` -> "0001, 0002, 0003, 0004, 0005, ..."
`$PrnCounter ("1234", Mode=7, ResetTime="06:00", ResetValue="0001")` -> `=CN(10;7;4;+1;1;06:00;0001)1234` -> "1234, 1235, 1236, 0001, 0002, ..."

See also

- Numeric Counter (Printer)
- Counter (System)

Numeric Counter

Defines a printer-internal numeric counter.

Syntax

```
$PrnCounterExt (start_value, [Prompt=prompt_text], [UpdateInterval=update_interval],
[Increment=increment], [MinValue=min_value], [MaxValue=max_value], [TrimLeft=trim_left],
[Mode=mode], [UpdateValue=update_value])
```

Parameters

start_value
Current start value.
Note: The number of digits of the start value determines the output format. A maximum of 9 digits can be displayed.

prompt_text (optional)
If a prompt text is defined, the start value must be entered at print start.

update_interval (optional, default = 1)
Indicates how often the variable is to be updated during a print job.
1: After each label
n: After n labels

increment (optional, default = 1)
Value to increment or decrement the counter.

min_value (optional, default = "0")
Minimum value

max_value (optional)
Maximum value. If no maximum value is specified, the number of digits of the start value is used by default to calculate the maximum value.

Start Value	Calculated Max Value
0001	9999
01	99

trim_left (optional, default = 0)
Specifies whether leading zeros should be displayed or not.
0: Show leading zeros in the output
1: Suppress leading zeros in the output

mode (optional, default = 5)
Operating mode
0: Keep start value
1: Reset start value at print start
2: Enter start value at printer
3: Enter start value (= last end value) at printer
4: Reset start value at cycle end
5: Limit reached, reset to min/max
6: Limit reached, reset to start value
7: Limit reached, stop printing

update_value (optional, only relevant for operation mode 0, 1, 4, 5, 6 and 7, default = 0)
Specifies whether the start value on the screen should be overwritten or not.
0: Keep start value
1: Overwrite start value

Examples

`$PrnCounterExt ("9995", Mode=5, Increment=1, UpdateInterval=1, MinValue="1", MaxValue="9999") -> =CC(+1,1,5,0,0,9999)0050` -> "9995, 9996, 9997, 9998, 9999, 0001, 0002, ..."
`$PrnCounterExt ("0050", Mode=5, Increment=1, UpdateInterval=1, MinValue="1", MaxValue="0055", TrimLeft=1) -> =CC(+1,1,5,0,1,0055)0050` -> "50, 51, 52, 53, 54, 55, 1, 2, 3, ..."

See also

- Counter (Printer)
- Counter (System)

Date/Time

Defines a printer-internal date and time variable.

Syntax

```
$PrnDateTime (output_format, [UpdateInterval=update_interval], [Mode=mode],
[MonthOffset=month_offset], [DayOffset=day_offset], [MinOffset=minute_offset],
[CorrectMonth=correct_month])
```

Parameters

output_format
Defines the output format to display date and time values on the label. For more information, see Printer-specific Date and Time Format Strings.

update_interval (optional, default = 0)
Indicates how often the variable is to be updated during a print job.
0: At print start
1: After each label

mode (optional, default = 0)
Specifies whether the start date should be entered at the printer or not.
0: Use current printer date
1: Enter start date (= temporary printer date) at print start

month_offset (optional, default = 0)
Month offset (added to the start date)

day_offset (optional, default = 0)
Day offset (added to the start date)

minute_offset (optional, default = 0)
Minute offset (added to the start time)

correct_month (optional, default = 0)
Month correction
0: Change into next month
1: Retain current month

Examples

Printer settings: 2/15/2014 14:21:25

```
$PrnDateTime ("YYMODD") -> =CL(0;0;0;0;0)<YYMODD> -> "140225"
$PrnDateTime ("HH:MI:SS", UpdateInterval=1, MinOffset=-60) -> =CL(0;0;1;-60;0)<HH:MI:SS> -> "13:21:25"
```

See also

- Date/Time (System)

Printer-specific Date and Time Format Strings

These format strings are used to define the text representation of a Date/Time (Printer) variable.

Custom Format Strings for Date

Format String	Description	Range	Examples
DD	Day of month	01-31	01.06.2009 13:45:30 -> 01 15.06.2009 13:45:30 -> 15
DOY	Day of year (julian day)	001-366	15.06.2009 13:45:30 -> 166
DY	Day of year	000-365	15.06.2009 13:45:30 -> 165
DW	Numerical day of week, where Sunday is 0 and Saturday is 6.	0-6	15.06.2009 13:45:30 -> 1
DW1	Numerical day of week, where Sunday is 1 and Saturday is 7.	1-7	15.06.2009 13:45:30 -> 2
Dwx	Day of week For x any ASCII character can be used, from this is counted consecutively started with Sunday.		15.06.2009 13:45:30 -> B (DwA)
DOWxxxxxxx	Day of week (variable) For x any ASCII character can be used. The first 'x' stands for Sunday, the next for Monday etc. until Saturday. **Note:** For each weekday a sign must be indicated.		
MO	Month	01-12	15.06.2009 13:45:30 -> 06
WW	Calendar week	01-53	15.06.2009 13:45:30 -> 25
Y	Year (1 digit)	0-9	15.06.2009 13:45:30 -> 9
YY	Year (2 digits)	00-99	15.06.2009 13:45:30 -> 09
YYYY	Year (4 digits)		15.06.2009 13:45:30 -> 2009

Custom Format Strings for Time

Format String	Description	Range	Examples
HH	Hour (24-hour clock)	00-23	15.06.2009 01:45:30 -> 01 15.06.2009 13:45:30 -> 13
HE	Hour (12-hour clock)	01-12	15.06.2009 01:45:30 -> 01 15.06.2009 13:45:30 -> 01
MI	Minutes	00-59	15.06.2009 01:09:30 -> 09 15.06.2009 13:09:30 -> 09
SS	Seconds	00-59	15.06.2009 13:45:09 -> 09
AM	AM/PM indicator	AM or PM	15.06.2009 13:45:09 -> PM
Am		a.m. or p.m.	15.06.2009 13:45:09 -> p.m.
am		am or pm	15.06.2009 13:45:09 -> pm

Culture-specific Format Strings

Format String	Description	Examples
xMO	Abbreviated month name	6/15/2009 01:45:30 PM -> JN (CMO) 6/15/2009 01:45:30 PM -> JUN (DMO) 6/15/2009 01:45:30 PM -> GUI (IMO)
xSO	Full month name	6/15/2009 01:45:30 PM -> June (ESO) 6/15/2009 01:45:30 PM -> Juin (FSO) 6/15/2009 01:45:30 PM -> Junio (SSO)
xSD	Abbreviated weekday name	6/15/2009 01:45:30 PM -> MO (GSD) 6/15/2009 01:45:30 PM -> MA (NSD) 6/15/2009 01:45:30 PM -> LUN (SSD)
xLD	Full weekday name	6/15/2009 01:45:30 PM -> Monday (ELD) 6/15/2009 01:45:30 PM -> Montag (GLD) 6/15/2009 01:45:30 PM -> Mandag (OLD)

For x the country abbreviation of the desired language can be used.

C = Canadian
D = Danish
E = English
F = French
G = German
I = Italian
N = Dutch
O = Norwegian
S = Spanish
U = Finnish
W = Swedish

Database Field

Defines a printer-internal database field.

Syntax

```
$PrnDbField (database_name, index_field, output_column, [Placeholder=placeholder_text])
```

Parameters

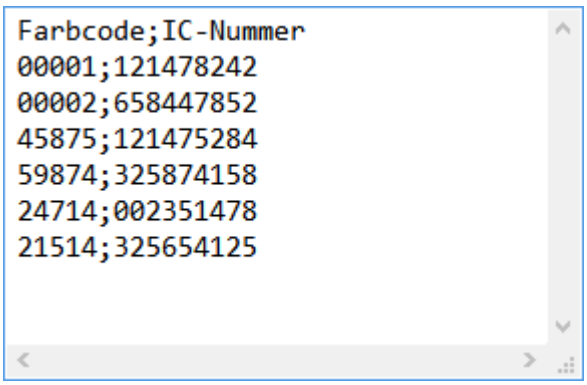
database_name
Database name
Note: Only data connections of type "Text File (Printer)" can be used.

index_field
The name of the field on the label, the content of which should be used for the data set search in the index column.
Note: Only printer-internal fields can be used.

output_column
Name or index of the column which should be output.
Note: Please take care to ensure that capitalisation is taken into account in the column names and the column index has a base of 1.

placeholder_text (optional)
Placeholder for on-screen display. *output_column* is displayed by default.

Examples




```
$PrnDbField ("Data", "Farbcode", "IC-Nummer") -> =MD(FN="U:\Standard
\Data.csv";SE=';';CH=1;SC="Farbcode",SF="Farbcode";RC="IC-Nummer")
```

Printout

Farbcode = "00001" -> "121478242"
Farbcode = "00002" -> "658447852"
Frabcode = "45875" -> "121475284"

An example of how to create a database label with printer-internal variables can be found here.

See also

- Create CSV File
- Save CSV File on Memory Card
- Database Field (System)

Create CSV File

CSV is a simple file format used to store tabular data. CSV stands for "comma-separated values". Its data fields are most often separated, or delimited, by a comma or a semicolon. For example, let's say you had a spreadsheet containing the following data.

Farbcode	IC-Nummer
00001	121478242
00002	658447852
45875	121475284
59874	325874158
24714	002351478
21514	325654125

This data could be represented in a CSV-formatted file as follows:

```
Farbcode;IC-Nummer
00001;121478242
00002;658447852
45875;121475284
59874;325874158
24714;002351478
21514;325654125
```

Here, the fields of data in each row are delimited with a semicolon and individual rows are separated by a newline.

A CSV is a text file, so it can be created and edited using any text editor. More frequently, however, is created by exporting (File Menu -> Export) a spreadsheet or database in the program that created it. Click on a link below for the steps to create a CSV file in Notepad and Microsoft Excel.

Notepad (or any text editor)
Microsoft Excel

See also

- CSV File Format

Notepad (or any text editor)

To create a CSV file with a text editor, first choose your favorite text editor, such as Notepad, and open a new file. Then enter the text data you want the file to contain, separating each field with a semicolon and each row with a new line.

```
Column1;Column2;Column3
one;two;three
example1;example2;example3
```

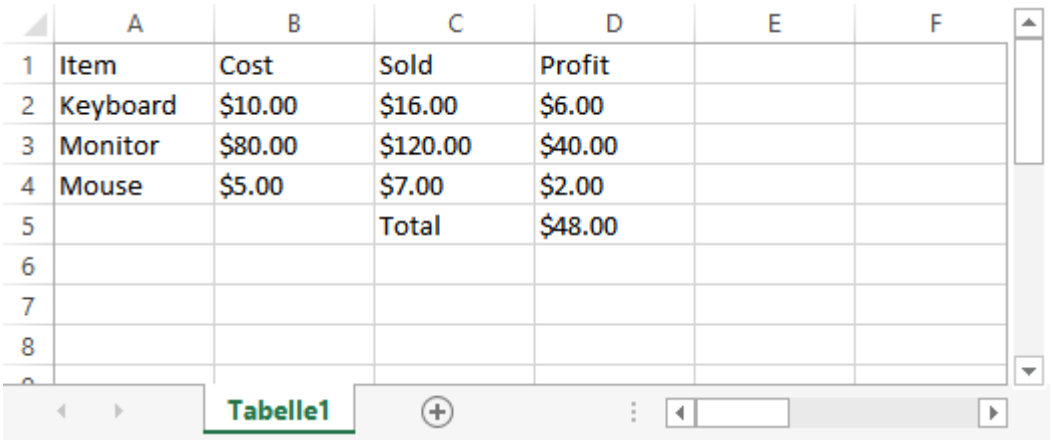
Save this file with the extension **.csv**.

See also

- Create CSV file

Microsoft Excel

To create a CSV file using **Microsoft Excel**, launch **Excel** and then open the file you want to save in CSV format. For example, below is the data contained in our example Excel worksheet:



	A	B	C	D	E	F
1	Item	Cost	Sold	Profit		
2	Keyboard	$10.00	$16.00	$6.00		
3	Monitor	$80.00	$120.00	$40.00		
4	Mouse	$5.00	$7.00	$2.00		
5			Total	$48.00		
6						
7						
8						

Tabelle1

Once open, click **File**, choose the **Save As** option, and for the **Save as type** option, select **CSV (Comma delimited)**.

After you save the file, you are free to open it up in a text editor to view it or to edit it manually. Its contents will resemble the following:

```
Item;Cost;Sold;Profit
Keyboard;$10.00;$16.00;$6.00
Monitor;$80.00;$120.00;$40.00
Mouse;$5.00;$7.00;$2.00
;;Total;$48.00
```

Note: You'll notice that the last row begins with two semicolons. This is because the first two fields of that row were empty in our spreadsheet. Don't delete them... the two semicolons are required so that the fields correspond from row to row. They cannot be omitted.

See also

- Create CSV file

CSV File Format

Here are the rules of how data should be formatted in a CSV file, from the IETF's document, RFC 4180. In these examples, "CRLF" is used to represent a carriage return and a linefeed (which together constitute a newline).

- Each record (row of data) is to be located on a separate line, delimited by a line break. For example:

```
aaa;bbb;ccc
zzz;yyy;xxx
```

- There may be an optional header line appearing as the first line of the file with the same format as normal record lines. This header will contain names corresponding to the fields in the file and should contain the same number of fields as the records in the rest of the file. For example:

```
column1;column2;column3
aaa;bbb;ccc
zzz;yyy;xxx
```

- Within the header and each record, there may be one or more fields, separated by semicolons. Each line should contain the same number of fields throughout the file. Spaces are considered part of a field and should not be ignored. The last field in the record must not be followed by a semicolon.

See also

› Create CSV file

Save CSV File on Memory Card

In order to save a CSV file on the memory card, proceed as follows:

1. Click on **Upload File** on the **Tools** tab in the **Memory Card** group.
2. Select the printer and the source file (e.g. on the hard disk).
3. Enter the pathnames for the target file on the Memory Card.
4. Click **OK** in order to save the file on the Memory Card.

Sample

Example

To learn how you can create a database label and to see just how easy it is, click on this link to see our video tutorials.
The sample data can be found in the installation folder in the *Samples\Database (Printer)* directory or can be downloaded from here.

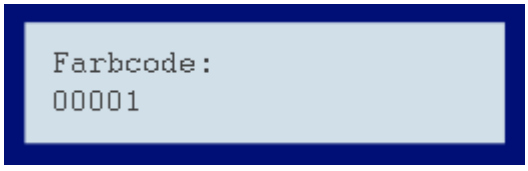
Data.csv This file contains the data in CSV format.
Import Data_csv data connection.lbdx Import file for data connection *Data*.
Label.lbex Sample label

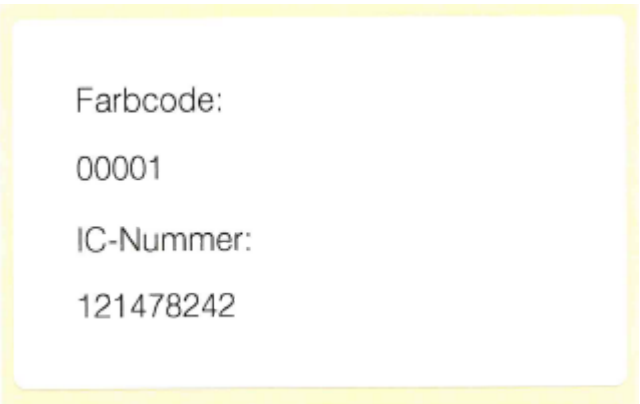
To print the sample label, proceed as follows:

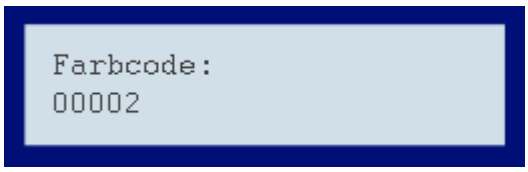
1. Open **Labelstar Office**.
2. Import data connection **Data**.
   - Activate **Data Connections** view.
   - Right-click to open context menu and click **Import Data Connections**.
   - Navigate to samples folder and open **Import Data_csv data connection.lbdx**.

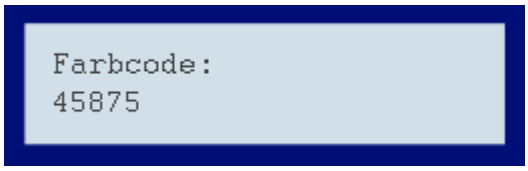
Farbcode	IC-Nummer
00001	121478242
00002	658447852
45875	121475284
59874	325874158
24714	002351478
21514	325654125

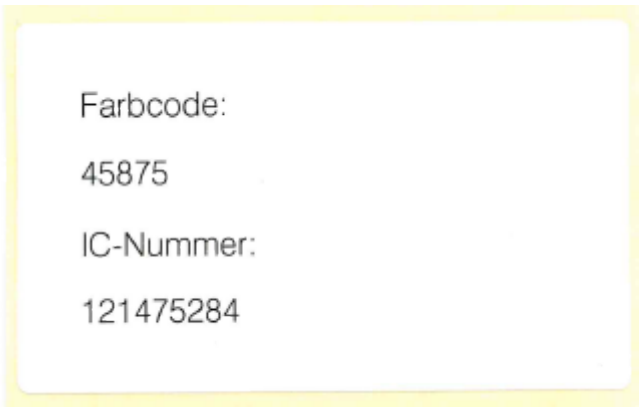
3. Save file on Memory Card
   - Expand data connection **Data** and click on **Properties**.
     The **Properties** dialog box opens.
   - If necessary, change the path in **Memory Card File**.
   - Click **Save File on Memory Card**.
4. Open and print sample label











Field Link

Defines a printer-internal field link.

Syntax

```
$PrnFieldLink (text_or_field_name, ...)
```

Parameters

text_or_field_name
Text constant or field name. A text constant must be placed into quotation marks ("). The quotation marks are not printed.
Note: Only printer-internal fields can be used.

Examples

ID01 = "12345"
ID02 = "67890"

`$PrnFieldLink (ID01, " - ", ID02)` -> `SC=(0;" - ";1)` -> "12345 - 67890"

See also

- Field Link (System)

Substring

Defines a printer-internal substring.

Syntax

```
$PrnSubstring (text_or_field_name, [start_index], [length])
```

Parameters

text_or_field_name
Text constant or field name. A text constant must be placed into quotation marks ("). The quotation marks are not printed.
Note: Only printer-internal fields can be used.

start_index (optional, default = 1)
The position of the first character to be returned.

length (optional)
The number of characters to be returned. If omitted all characters from the starting position to the end of the string are returned.

Examples

ID01 = "123456789"

```
$PrnSubstring (ID01) -> =SS(0;1) -> "123456789"
$PrnSubstring (ID01, 4) -> =SS(0;4) -> "456789"
$PrnSubstring (ID01, 4, 2) -> =SS(0;4;2) -> "45"
```

See also

- Substring (System)

User Input

Defines a printer-internal user input.

Syntax

```
$PrnUserInput (prompt_text, start_text, [AllowableChars=allowable_chars],
[SkipSpecialChars=skip_special_chars], [RightAlignText=right_align_text], [InputMask=input_mask],
[MaxLength=max_length], [PadChar=pad_char])
```

Parameters

prompt_text
The prompt text shown in the first line of the printer display.

start_text
The input text shown in the second line of printer display.

allowable_chars (optional, default = Alphanumeric)
Indicates which characters are allowed for the input.
Numeric|0: Digits only
Alphanumeric|1: Digits and letters
InputMask|2: Input mask

skip_special_chars (optional, only relevant for numeric and alphanumeric user input, default = 0)
Indicates whether special characters are to be retained at the input or not.
0: Do not skip special characters
1: Skip special characters

Special characters are: point (.), colon (:), comma (,), semicolon (;), minus sign (-) and slash (/).

right_align_text (optional, only relevant for numeric and alphanumeric user input, default = 0)
Input alignment
0: Left-aligned
1: Right-aligned

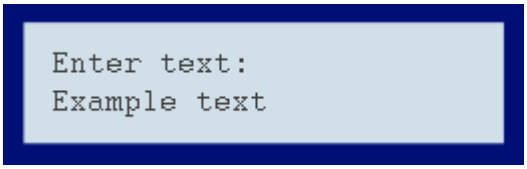
input_mask (optional, only relevant for user input with input mask)
Input mask. You can use the input mask to define a specific input format for user inputs. For more information, see Printer-internal Input Format Strings.

max_length (optional, only relevant for numeric and alphanumeric user input, default = 1)
The maximum number of characters that will be accepted as input on the printer. If *max_length* is less than the number of characters in *start_text*, the maximum number of characters that can be entered is the number of characters in *start_text*.

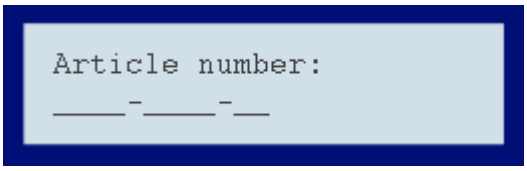
pad_char (optional, only relevant for numeric and alphanumeric user input, default = space)
The character that is used to pad *start_text* if the text length is less than *max_length*. Space is used by default.

Examples

```
$PrnUserInput ("Enter text:", "Example text", AllowableChars=1) -> =UG(0;1;0;0;1;"Enter
text:")<Example text>
```




```
$PrnUserInput ("Article number:", "", AllowableChars=2, InputMask="9999-9999-99") ->
=UM(0;0;0;0;1;"Article number:";1;"9999-9999-99")<>
```



See also

- User Input (System)

Printer-specific Input Format Strings

This format strings are used to define the input mask of a User Input (Printer).

You can use input masks to ensure that data is always entered correctly. This can range from phone numbers to postal codes to serial numbers, and much more.

The following table describes the wildcard characters that can be used in the input mask:

Character	Description
9	The user can enter a number (0-9).
#	The user can enter a number, a plus or a minus sign.
?	The user can enter a letter.
a	The user can enter a letter or a number.
C	The user can enter any character.

Examples of Inputs Masks

The examples in the following table illustrate some uses of input masks.

Input Mask	Usage	Input	Output
(000) 000-aaaa	Phone number (US format)	2065558353 206555TELE	(206) 555-8353 (206) 555-TELE
#999	Positive or negative number	-20 2000	-20 2000
??00000-0000	Article or serial number	DB513920493	DB51392-0493

SAPscript Variable

Defines a printer-internal SAPscript variable.

Syntax

```
$SAPField (field_name, [placeholder_text])
```

Parameters

field_name
SAPscript field name such as "VBAK-KUNNR".

placeholder_text (optional)
Placeholder for on-screen display. The field name is displayed by default.

Examples

	On-Screen Display	Printout
`$SAPField ("VBAK-KUNNR")`	&VBAK-NUNNR&	&VBAK-NUNNR&
`$SAPField ("&VBAK-KUNNR&", "12345")`	12345	&VBAK-NUNNR&

See also

- Printing in an SAP environment

User-defined Variables

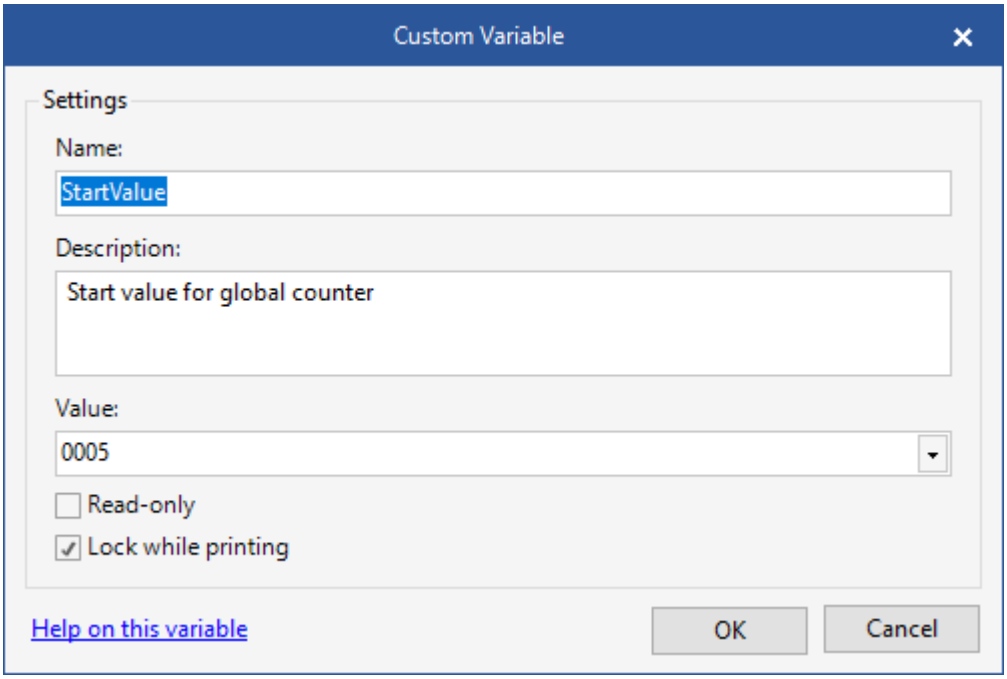
Labelstar Office supports user-defined variables.

Custom variables can be used in many ways in **Labelstar Office**. For example to define the file path of a data connection or to define a global counter.

To manage user-defined variables, proceed as follows:

1. Activate **Variables** view and expand **User-defined Variables**.
2. **Create new variable**
   - Click **New Variable**

     The **Custom Variable** dialog box opens.



3. **Edit variable**

   In order to edit a variable, perform one of the following actions:
   - Right-click on the variable and select the **Edit** context menu option.
   - Double-click on the variable which you want to edit.
4. **Delete variable**

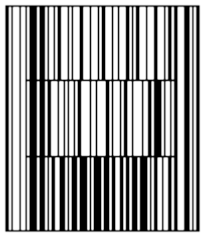
   In order to delete a variable, perform one of the following actions:
   - Right-click on the variable and select the **Delete** context menu option.
   - Select the variable which you want to delete and press DEL.

See also

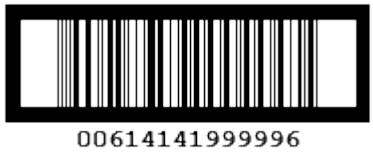
- Share programs

Barcodes

Following is a list of supported bar code types:

Code	Example	Description
Aztec Code		2D barcode, developed by Welch Allyn.
Aztec Runes		2D barcode based on the Aztec Code.
Codabar		Numeric barcode, encoded digits and special characters.
Codablock F		2D barcode based on the Code 128.
Code 128		Alphanumeric barcode, encoded ASCII character set.
Code 2/5 Industrial		Numeric barcode.
Code 2/5 Interleaved		Numeric barcode, with an even number of digits.
Code 39		Alphanumeric barcode, encoded digits, capital letters, special characters and spaces.
Code 39 Extended		Alphanumeric barcode based on Code 39, encoded full ASCII character set.

Code 93		Alphanumeric barcode, encoded digits, capital letters, special characters and spaces.
Code 93 Extended		Alphanumeric barcode based on Code 93, encoded full ASCII character set. **Note:** This barcode is transmitted graphically.
DataMatrix		2D barcode, developed by Acuity Corp.
Deutsche Post Identcode		Numeric postal barcode based on Code 2/5 Interleaved with different check digit calculation.
Deutsche Post Leitcode		Numeric postal barcode based on Code 2/5 Interleaved with different check digit calculation.
EAN-13, GTIN-13		Numeric barcode.
EAN-13 + 2 Stellen		EAN-13 with 2-digit add on.
EAN-13 + 5 Stellen		EAN-13 with 5-digit add on.
EAN-8, GTIN-8		Numeric barcode.
GS1-128		Alphanumeric barcode based on the Code 128.
GS1 DataBar		Alphanumeric barcode.

GS1 DataMatrix		2D barcode.
HIBC-128		Alphanumeric barcode based on the Code 128.
HIBC-39		Alphanumeric barcode based on Code 39.
HIBC DataMatrix		2D barcode.
HIBC QR Code		2D barcode.
ITF-14, SCC-14		Numeric barcode based on Code 2/5 Interleaved.
MaxiCode		2D barcode.
PDF417		2D barcode.
Pharmacode		Numeric barcode.
PZN		Numeric barcode based on Code 39.

QR Code		2D barcode.
UPC-A, GTIN-12		Numeric barcode.
UPC-E		Numeric barcode.
USPS Intelligent Mail® Barcode		Numeric postal barcode.
USPS PostNet		Numeric postal barcode.

1D-Barcodes

Linear (one dimensional) barcodes consist of a single row of parallel bars and spaces of varying widths that represent data. The bars and spaces are arranged in a predetermined pattern defined by the symbology.

Supported Barcodes

- Codabar
- Code 128
- Code 2/5 Industrial
- Code 2/5 Interleaved
- Code 39
- Code 39 (Full ASCII)
- Code 93
- Code 93 (Full ASCII)
- EAN-13
- EAN-8
- GTIN-8
- GTIN-12
- GTIN-13
- ITF-14
- Pharmacode
- PZN
- SCC-14
- UPC-A
- UPC-E

Codabar

The **Codabar** is mainly used in libraries, in photo sector and in medical ranges (blood banks). The **Codabar** is a universal, numeric barcode that contains 6 special characters additionally to the numbers 0 to 9. The number of representable signs is not given of the code.

Additionally four different start/stop signs (A-D) are defined, i.e. each code must begin and end with A, B, C or D. However, the start/stop signs cannot be used in the bar code itself.

Each sign of the code consists of elf units, four bars and three spaces. A fourth gap is always narrow.

Technical Specifications

Length	Variable
Valid characters	Digits 0-9 Special characters - $: / . +
Check digit	Optional Modulo 16

Code 128

Code 128 is a universal, alphanumeric barcode mainly used in shipping/transport, on documents of identification and in warehousing/distribution.

Code 128 can encode the complete ASCII character set. This code uses an internal check digit that won't be displayed in the text line under the code. By the use of four different widths for bars and gaps. the information density is very high.

The structure of a **Code 128** consists of a start sign, data area, check digit and a stop sign. Before the start sign and behind the stop sign a white zone (quiet zone) with a width of at least 10 modules must be defined.

Internal Character Representation

Although the Code 128 only knows 102 different characters, all 128 ASCII characters can be displayed. For this purpose, the system automatically switches back and forth between the character sets A, B and C in order to encrypt alphanumeric data in the shortest possible form.

- **Character Set A:** Digits, capital letters, control and special characters
- **Character Set B:** Digits, letters, special characters
- **Character Set C:** 00-99 (encodes two digits with a single character)

Technical Specifications

Length	Variable
Valid characters	ASCII character set including control characters
Check digit	Modulo 103

See also

- GS1-128

Code 2/5 Industrial

The **Code 2 of 5 Industrial** is a very simple numeric code which is able to display digits from 0 to 9. The code is mainly used in industrial sector and particularly in transport and warehousing. **Code 2 of 5** has no built in check digit.

As the information density of the bar code is low and its space consumption very high, it is barely used nowadays.

The bar code has its name because each number is coded in 5 bars, two broad bars and three narrow bars. The spaces between the bars not contain any information.

Technical Specifications

Length	Variable
Valid characters	Digits 0-9
Check digit	Optional Modulo 10 Modulo 10 (Luhn Algorithm)

See also

- Code 2/5 Interleaved

Code 2/5 Interleaved

Code 2 of 5 Interleaved is a special type of Code 2 of 5 Industrial that is also a numeric code able to display digits from 0 to 9. The advantage of **Code 2 of 5 Interleaved** is that the code uses self-checking and it is very compact so it does not need much space like the simple Code 2 of 5 Industrial.

Code 2 of 5 Interleaved is only valid if there is a even number of digits. To display an odd number of digits you have to add a zero to the beginning (123 becomes 0123) or you may use your own check digit.

Technical Specifications

Length	Variable (even number of digits)
Valid characters	Digits 0-9
Check digit	Optional Modulo 10 Modulo 10 (Luhn Algorithm)

See also

- Code 2/5 Industrial

Code 39

Code 39 is an alphanumeric code mainly used in shipping/transport, electronics and chemical industries, the health sector and and in warehousing/distribution.

Each character is composed of nine elements: five bars and four spaces. Three of the nine elements in each character are wide (binary value 1), and six elements are narrow (binary value 0). The width ratio between narrow and wide is not critical, and may be chosen between 1:2 and 1:3. The barcode itself does not contain a check digit, but it can be considered self-checking on the grounds that a single erroneously interpreted bar cannot generate another valid character.

Possibly the most serious drawback of **Code 39** is its low data density: It requires more space to encode data in **Code 39** than, for example, in Code 128.

Technical Specifications

Length	Variable
Valid characters	Digits 0-9 Capital letters A-Z Special characters - . $ / + % Space
Check digit	Optional Modulo 43 Modulo 11 (weighting 7) Modulo 10 (Luhn Algorithm)

See also

- Code 39 Extended

Code 39 Extended

Code 39 Extended is an extended version of Code 39 that can encode the complete ASCII character set.

The additional characters (e.g. lower case letters) are created using the existing characters of Code 39 by combining two characters each.

Technical Specifications

Length	Variable
Valid characters	ASCII character set
Check digit	Optional Modulo 43 Modulo 11 (weighting 7) Modulo 10 (Luhn Algorithm)

See also

- Code 39

Code 93

Code 93 is an alphanumeric code similar to Code 39 and can encode 48 different characters.

By the use of various bar widths and gap widths it has a higher information density.

Each sign of the code consists of nine units, three bars and three spaces.

Technical Specifications

Length	Variable
Valid characters	Digits 0-9 Capital letters A-Z Special characters - . $ / + % Space
Check digit	Modulo 47

See also

- Code 93 Extended

Code 93 Extended

Note

This barcode is transmitted graphically.

Code 93 Extended is an extended version of Code 93 that can encode the complete ASCII character set.

Technical Specifications

Length	Variable
Valid characters	ASCII character set
Check digit	Modulo 47

See also

- Code 93

EAN-13, GTIN-13

EAN-13 is used world-wide for marking retail goods. Each packaging is uniquely identified by the GTIN - Global Trade Item Number (formerly European Article Number - EAN).

The symbol encodes 13 characters: the first two or three are a country code which identify the country in which the manufacturer is registered (not necessarily where the product is actually made). The country code is followed by 9 or 10 data digits (depending on the length of the country code) and a single check digit.

Add Ons

In addition to the user data, it is possible to define a 2- or 5-digit additional code (AddOn) for this barcode type.

Example: ISBN 999-3-12345-123-5 with price code (55.00 EUR)

Technical Specifications

Length	13
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- EAN-8, GTIN-8

EAN-8, GTIN-8

EAN-8 is a shortened version of the EAN-13 code.

It includes a 2 or 3 digit country code, 4 or 5 data digits (depending on the length of the country code), and a check digit.

A GTIN short number will only be output upon request since these numbers have limited availability. The **EAN-8 barcodes** with a 2 as the starting digit can be used freely within the company, but they are not unique in the world.

Technical Specifications

Length	8
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- EAN-13, GTIN-13

ITF-14, SCC-14

The **ITF-14**, which is based on Code 2 of 5 Interleaved, is used to create the Shipping Container Code (SSC). This code is used to mark cartons and palettes that are including goods with an EAN-13 code.

A **SCC-14** number contains the following information:

- **1:** Package indicator
- **2..3:** UPC numbering system/GS1 country prefix
- **4..8:** GS1 company prefix
- **9..13:** Item identification number
- **14:** Check digit

Technical Specifications

Length	14
Valid characters	Digits 0-9
Check digit	Modulo 10

Pharmacode

The **Pharmacode** is a simple, numeric barcode placed on the marked from company Laetus. It is used in pharmaceutical industry for the control of packaging means and/or for the control of packaging machines.

The **Pharmacode** applied on the packaging and on the package insert provides that the correct package insert is sorted into the appropriate packaging. With the **Pharmacode** only integers can be coded from 3 to 131070.

Technical Specifications

Length	Variable
Valid characters	Digits 0-9
Check digit	None

PZN

The **PZN** (Pharmazentralnummer) serves for marking of drugs and other pharmacy products according to trademarks, dosage form, intensity and package size.

The **PZN** is assigned by the Informationsstelle für Arzneispezialitäten (IFA).

The **PZN-8** replaces the old **PZN-7** from the 01.01.2013. You will be able to convert old **PZN-7** code to **PZN-8** by just adding a leading zero.

Technical Specifications

Length	7-8
Valid characters	Digits 0-9
Check digit	Modulo 11

UPC-A, GTIN-12

The Universal Product Code (UPC) is a barcode symbology that is widely used in the United States, Canada, the United Kingdom, Australia, New Zealand and in other countries for tracking trade items in stores.

Its most common form, the **UPC-A**, consists of 12 numerical digits, which are uniquely assigned to each trade item. Along with the related EAN-13 barcode, the **UPC-A** is the bar code mainly used for scanning of trade items at the point of sale.

The symbol encodes 12 characters:

- **1:** System identification
- **2..6:** UPC ID number (manufacturer)
- **7..11:** Individual article number (issued by the manufacturer)
- **12:** Check digit

Technical Specifications

Length	12
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- UPC-E

UPC-E

The **UPC-E** is intended to be used on packaging which would be otherwise too small to use one of the other versions. The code is smaller because it drops out zeros which would otherwise occur in the symbol. For example, the code 59300-00066 would be encoded as 593663. The last digit (3 in the example) indicates the type of compression.

Technical Specifications

Length	8
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- UPC-A

2D Barcodes

Most 2D barcodes consist of small black and white squares and encode information in the area. A distinction is made between stacked barcodes, matrix codes, item codes and other special shapes.

Supported Barcodes

- Aztec Code
- Aztec Runes
- Codablock F
- DataMatrix
- MaxiCode
- PDF417
- QR Code

Aztec Code

Aztec Code is a 2D matrix code which is built on a square grid with a bulls-eye pattern at its centre for locating the code. In the concentric square rings around the bulls-eye pattern data is encoded. **Aztec Code** is used mainly in transportation e.g. for online tickets of Deutsche Bahn.

Very small (starting from 12 characters) and large data volumes (up to 3067 alphanumeric characters) can be coded.

Aztec Code consists of three fix and two variable components. The fix components are: the central Finder Pattern, Orientation Pattern and Reference Grid. Mode Message and Data Layers are the variable components of the code.

The **Aztec Code** is one of the few bar codes which do not need a quiet zone. Thanks to the Reed-Solomon error correction the reconstruction of data contents is still possible even if the code (25% with large codes and 40% with small codes) was destroyed. The so-called Core Symbol of the **Aztec Code** contains the central Finder Pattern, Orientation Pattern and Mode Message.

Technical Specifications

Length	3067 alphanumeric characters 3832 numeric characters
Valid characters	ASCII character set
Check digit	Internal

See also

- Aztec Runes

Aztec Runes

Aztec Runes are a set of small barcode symbols that are used for special applications.

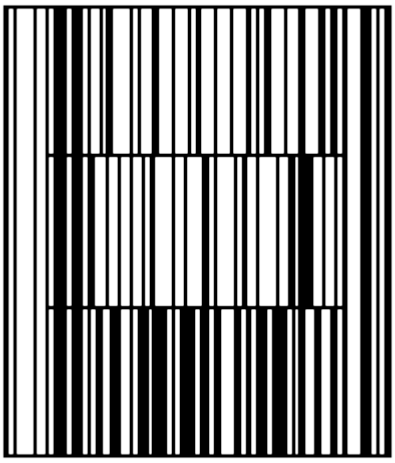
Technical Specifications

Length	3
Valid characters	An integer between 0 and 255 (including the boundaries).
Check digit	Internal

See also

- Aztec Code

Codablock F



Codablock F is a 2D barcode with several stacked Code 128 one above the other. The code is mainly used in health care.

The lines of **Codablock F** are marked by line numbers exactly as the total character number of the code. With **Codablock F** 2 to 44 lines can be displayed. Each individual line can have up to 62 characters according to Code 128.

The principle of Codablock barcode is the same like the line break of a text editor - if the line is full it is wrapped to the next line, i.e. to each line the line number is added and to the finished block the number of lines. For the orientation of the reading device, each line contains a line indicator and additionally two check digits to secure the contents of the total message.

Technical Specifications

Length	In 2 to 44 lines each, 4 to 62 characters (max 2725 characters) are encoded.
Valid characters	ASCII character set
Check digit	Internal

DataMatrix

DataMatrix code is one of the most popular 2D barcodes. Many data can be coded onto a small surface. For this reason it is often used for permanent marking with lasers in production (e.g. circuit boards). Additionally the code is used in the automotive sector, with analyzers and instruments (chemistry, medicine) and also increasingly as printed code image in documentation (e.g. tickets, digital postmarks).

DataMatrix symbols consist of square modules arranged within black (active ->binary 1) and white (inactive -> binary 0) cells.

Finder pattern as area of **DataMatrix** code has the width of a module and consists at the left and upper sides of two lines, which contain only dark modules. The lines at the right and upper side of the symbol are represented alternating in dark and bright modules. A quiet zone with the width of one module surrounds the barcode.

The uniform symbol size and the firm symbol distance make reading and decoding of the code very safe. A **DataMatrix** barcode symbol consists of the following four components:

- **Data area:** This area contains redundant data in codified form for data protection.
- **Closed limitation line (finder pattern):** This is the corner that is represented in normal alignment to the left and below data area with an uninterrupted line. This boundary is used for erection and rectification of the code in order to permit each reading angle.
- **Open borderline (alternating pattern):** This represents the opposite corner of the 'closed limitation line'. These lines are on top and right sides and consist of white and black dots (open lines). These are used to the determination of lines and columns while scanning.
- **Quiet zone:** Zone around the code containing no information or pattern. This area must be at least so wide as one column/line res. one dot of the code.

For the creation of **DataMatrix** code the Reed-Solomon error correction code ECC 200 is used. With this error correction code a **DataMatrix** barcode is still readable even if up to 25% of the code is covered or destroyed.

Technical Specifications

Length	Depending on the symbol size, a **DataMatrix** code can encode up to 3116 numbers or up to 2335 characters.
Valid characters	ASCII character set
Check digit	Internal

See also

- Symbol sizes
- GS1 DataMatrix

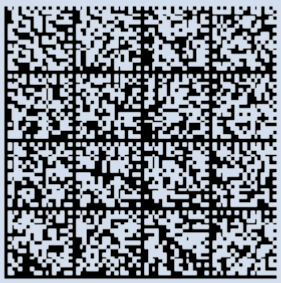
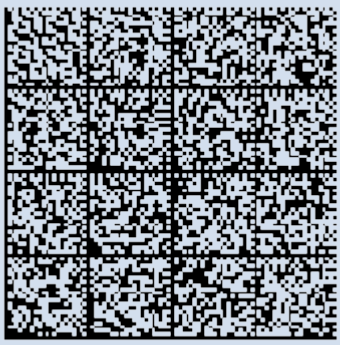
Symbol Sizes

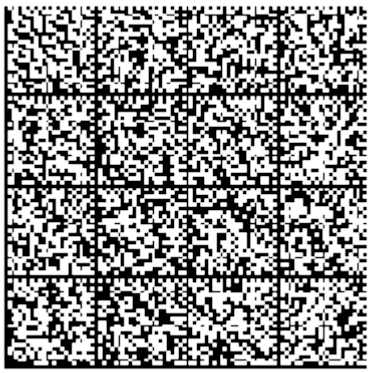
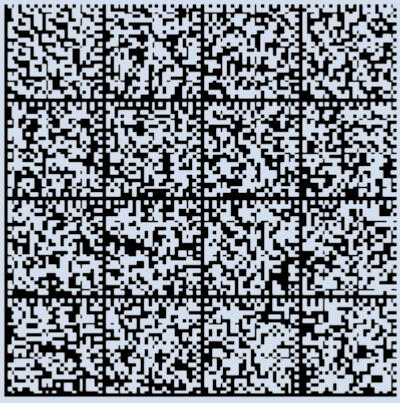
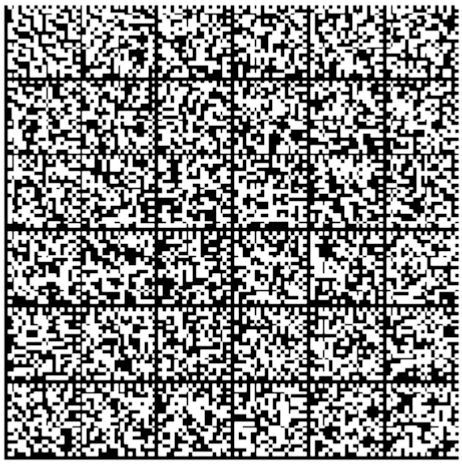
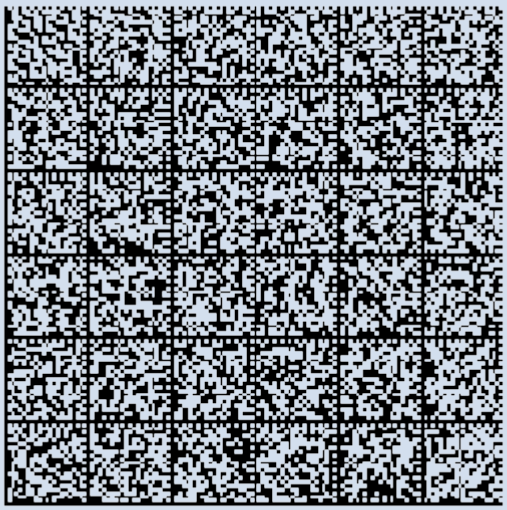
How many data fit into a DataMatrix code?

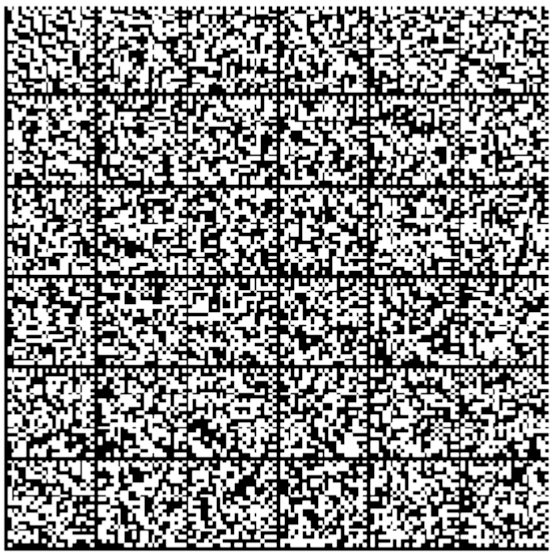
If symbol size is set as **Automatic Labelstar Office** always selects the best symbol size taht fits for the data to encode. The symbol sizes 12x12 and 8x18 and the symbol size 20x20 and 12x36 have the same number of data they can encode. **Labelstar Office** always uses the square version (i.e. 12x12 or 20x20) in this case.

Supported Symbol Sizes

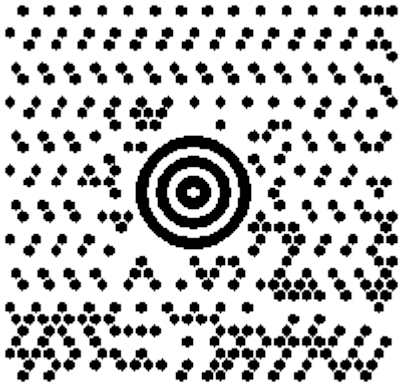
Symbol Size		Max Numeric Capacity	Max Alphanumeric Capacity	Max Correctable Error/Erasure
10 x 10		6	3	2/0
12 x 12		10	6	3/0
14 x 14		16	10	5/7
16 x 16		24	16	6/9
18 x 18		36	25	7/11
20 x 20		44	31	9/15
22 x 22		60	43	10/17
24 x 24		72	52	12/21
26 x 26		88	64	14/25
32 x 32		124	91	18/33
36 x 36		172	127	21/39
40 x 40		228	169	24/45

44 x 44		288	214	28/53
48 x 48		348	259	34/65
52 x 52		408	304	42/78
64 x 64		560	418	56/106
72 x 72		736	550	72/132
80 x 80		912	682	96/180
88 x 88		1152	862	112/212

96 x 96		1392	1042	136/260
104 x 104		1632	1222	168/318
120 x 120		2100	1573	204/390
132 x 132		2608	1954	248/472

144 x 144		3116	2335	310/590
8 x 18		10	6	3/0
8 x 32		20	13	5/0
12 x 26		32	22	7/11
12 x 36		44	31	9/15
16 x 36		64	46	12/21
16 x 48		98	72	14/25

MaxiCode



MaxiCode is a 2D barcode with a fixed size of 1 in. x 1 in. (approx. 25,4 mm x 25,4 mm). In this area of 1 square inch (approx. 645 mm^2) data can be coded. The code is build of 884 hexagonal modules that show finder pattern.

MaxiCode is a machine-readable symbol system created and used by United Parcel Service. The code is suitable for fast identification, tracking and managing the shipment of packages and contains the UPS control number, weight, kind of dispatch and address.

The code is easily identifiable at the bull's-eye pattern in the middle of the symbol. By the Reed-Solomon error correction a reconstruction of the 2D barcode is still possible even if up to 25% of the code were destroyed.

MaxiCode defines 6 modes that determines that how data should be interpreted. The mode 0 and 1 are no longer used. Mode 4 and 5 are used to encode "raw data" with mode 5 offers a slight higher data error correction. Mode 2 and 3 are used to encode “structure message” which comprises two parts: Primary Message and Secondary Message. The Primary Message encodes a postal code, 3-digit country code and 3-digit class of service code. The Second Message encodes other data.

Labelstar Office supports the following modes:

- **Mode 2:** Indicates the symbol contains a Structured Carrier Message with a numeric postal code (up to 9 digits).
- **Mode 3:** Indicates the symbol contains a Structured Carrier Message with an alphanumeric postal code (up to 6 characters).
- **Mode 4:** Indicates the symbols contains general-purpose data, protected by the standard error correction.

Technical Specifications

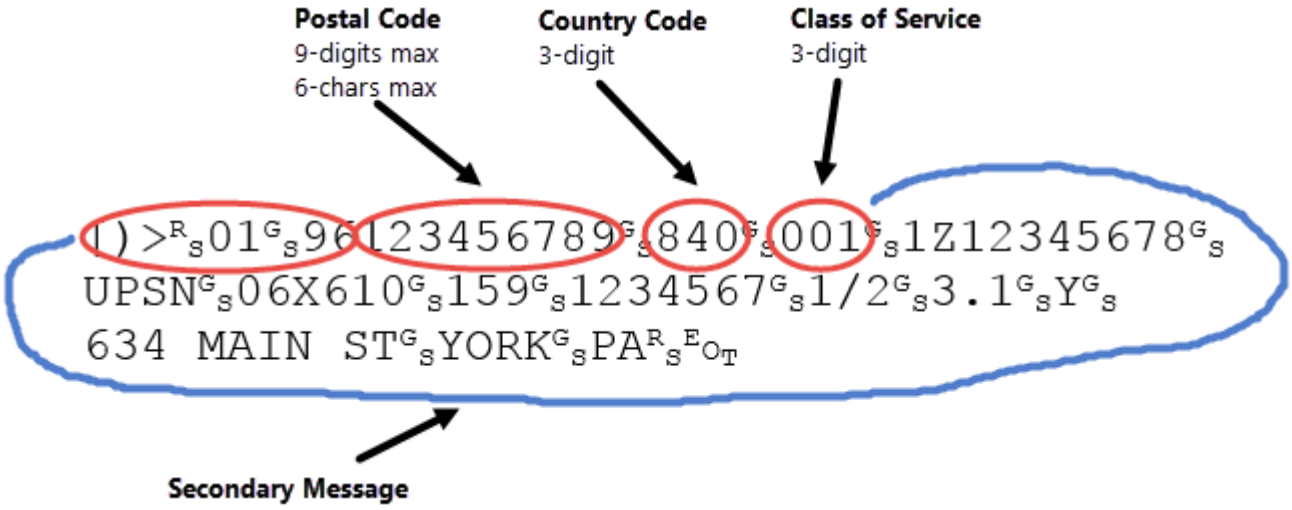
Length	93 alphanumeric characters 138 numeric characters
Valid characters	ASCII character set
Check digit	Internal

Structure Carrier Message

UPS has specific requirements of their customers for the content of **MaxiCode** symbols.

The message format that UPS uses in **MaxiCode** symbols conform to the ANSI MH10.8M-1993 standard. ANSI MH10.8M-1993 is an American National Standard for barcodes on unit loads and transport packages. This standard sets down guidelines on how package information can be coded so that it can be consistently and reliably exchanged between organizations. This record format is commonly referred to as a Structured Carrier Message.

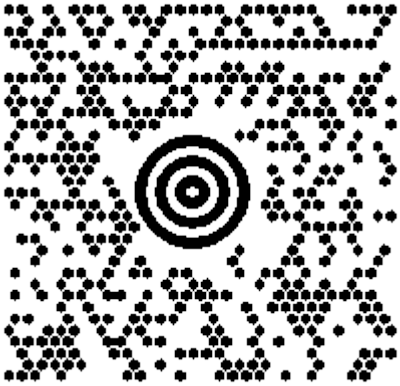
Message Structure



Field	Description	Size and Type	Required	Example
Primary Message				
Postal Code		5 or 9 digits in the USA (Mode 2), up to 6 alphanumeric charcters in other countries (Mode 3).	Yes	123456789
Country Code	Country code encoded per ISO 3166. Mode 2 supports the US Country Code (840). For other country codes please use Mode 3 instead.	3 numeric digits	Yes	840
Service Class	Assigned by the carrier	3 numeric digits	Yes	001
Secundary Message				
Tracking Number		10 or 11 alphanumeric characters, alpha characters must be uppercase	Yes	1Z12345678
Carrier Number	Standard Carrier Alpha Code	"UPSN"	Yes	UPSN
Shipper Number		6 alphanumeric characters, alpha characters must be uppercase	Yes	06x610
Julian Day of Pickup		3 numeric digits	Yes	156
Shipment Number		0-30 alphanumeric characters, alpha characters must be uppercase	-	1234567
Package n/x	Package n of x total packages.	1-3 numeric digits"/"1-3 numeric digits	Yes	1/2
Package Weight	Package weight in pounds.	1-3 numeric digits	Yes	3
Address Validation		"Y" or "N"	Yes	Y

Ship-To Address		0-35 alphanumeric characters, alpha characters must be uppercase	-	634 MAIN ST
Ship-To City		1-20 alphanumeric characters, alpha characters must be uppercase	Yes	YORK
Ship-To State		2 alpha characters, must be uppercase	Yes	PA

Example Code



PDF417

The **PDF417** is a stacked linear barcode based on a rectangular field. PDF stands for Portable Data File. It is used in a variety of applications, primarily identification cards, transport, automobile industry, inventory management and in administrative authority, e.g. Agentur für Arbeit to prevent manipulation at questionnaires.

The barcode symbol consists of 3 to 90 lines and 1 to 30 columns. Each line has a left and a right Quiet Zone, a Start/Stop Patterns, a left and a right indicator and 1 to 30 Symbol Characters. A **PDF417** symbol is formed of barcode data, check digit and correction sign. The used characters are coded in code words. A code word consists of 17 modules that are formed of 4 bars and spaces.

The error correction is determined with the Reed Solomon algorithm in 9 selectable Error Correction Levels. With selected error correction level 0 an error can be recognized but not corrected. With the error correction levels 1 to 8 errors can also be corrected.

Use of the error correction:

- **ECL 2:** less than 41 code words
- **ECL 3:** 41 to 160 code words
- **ECL 4:** 161 to 320 code words
- **ECL 5:** more than 320 code words

Technical Specifications

Length	1850 alphanumeric characters 2725 numeric characters
Valid characters	ASCII character set
Check digit	Internal

QR Code

A **QR Code** (quick response code) is a type of 2D barcode that is used to provide easy access to information through a smartphone.

In this process, known as mobile tagging, the smartphone's owner points the phone at a **QR Code** and opens a barcode reader app which works in conjunction with the phone's camera. The reader interprets the code, which typically contains a call to action such as an invitation to download a mobile application, a link to view a video or an SMS message inviting the viewer to respond to a poll. The phone's owner can choose to act upon the call to action or click cancel and ignore the invitation.

Technical Specifications

Length	4296 alphanumeric characters 7089 numeric characters
Valid characters	ASCII character set
Check digit	Internal

See also

- QR Codes and printer variables

What are the different types of QR Codes?

QR Codes can trigger various actions on the smartphone where they are read. Directing a user to a website isn't the only possible action and some of them are worth knowing (such as saving a business card or connecting to wireless networks).

With **Labelstar Office** you can create the following types of **QR Codes**:

- **Plain text:** This is the simplest QR Code type. A raw text is encoded and will be displayed on the screen after scanning. You can write anything you like.
- **Business card:** With these business card **QR Codes**, a contact card with the details you entered will be automatically stored into the contact list of the smartphone. You can enter your names, address, phone number, email and so on.
- **Add an event to a calendar:** After scanning these **QR Codes**, you will be asked if you want to save the event in your smartphone's calendar. By adding the event to your calendar, you will be reminded of the correct date.
- **Website:** This content type is probably the most used one. The **QR Code** just contains the address of a website. By scanning the code, the website can be access by the user without the hassle of manually entering the address (URL). This works because the QR reader recognizes the http(s):// protocol prefix and then interprets the text as a website address/URL.
- **Call a phone number:** Type in a phone number when you create the **QR Code**. When scanning, users will be proposed to call the phone number.
- **Send an SMS:** Save the content and the recipient's phone number of an SMS. After scanning, you will only have to confirm before sending it.
- **Send an email:** This works exactly like the SMS **QR Code** type. Only this time, you enter the email content, the subject and the recipients to enable sending after scanning.
- **Geo location:** Geographic co-ordinates are stored and when scanned will redirect to a static mobile google map of your location.
- **Wifi access information:** Whoever scans the code will be able to access your Wi-fi.

QR Codes and Printer Variables

Note

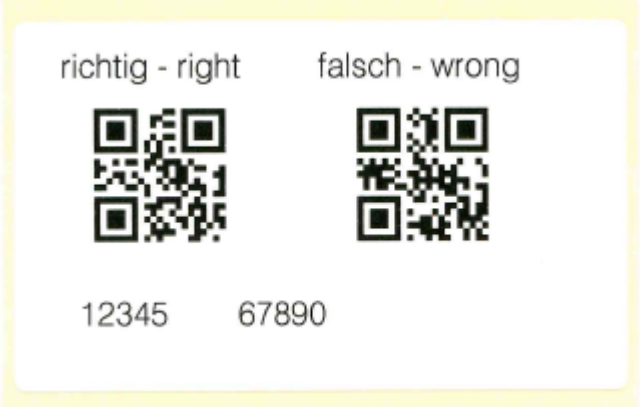
For label printers with a firmware Version 1.73 or older, if you want to use the **QR Code** with printer variables, the prefix "9999" must be added in order to print the barcode correctly. Newer firmware versions no longer need this prefix.

Example

The labels *QRCode_FieldLink.lbex* and *QRCode_Counter.lbex* can be found in the installation folder *Samples\QRCode* or can be downloaded here.

Examples

Right	Wrong
`$PrnFieldLink ("9999", Text1, Text2)` -> "1234567890"	`$PrnFieldLink (Text1, Text2)` -> "567890"



GS1 Barcodes

GS1 (Global Standard One) is a worldwide system, which facilitates unmistakable identification. Bar coded GS1 Application Identifiers for automated processing clearly label products/items, logistics units, reusable packaging/containers etc., as they are unique. Scanners then read the GS1 symbols error-free and process them. The GS1 keys form the basis for efficient and cost-effective goods flow management from the manufacturer to the end user.

Supported Barcodes

- GS1-128
- GS1 DataBar
- GS1 DataBar Composite
- GS1 DataMatrix

GS1-128

The **GS1-128** is a special form of Code 128. This barcode is used for goods and palettes mainly in commerce and industry. The name **GS1-128** replaces the old name EAN/UCC-128.

The length of **GS1-128** is variable, however should not exceed the maximum length of 165 mm. Altogether a maximum of 48 rated character including the Application Identifiers and FNC1 signs can be coded.

Technical Specifications

Length	Variable
Valid characters	ASCII character set
Check digit	Modulo 103

See also

- Code 128

GS1 DataBar

GS1 DataBar is a family of symbols most commonly seen in the GS1 DataBar Coupon. Formerly known as Reduced Space Symbology (RSS-14), this family of bar codes include:

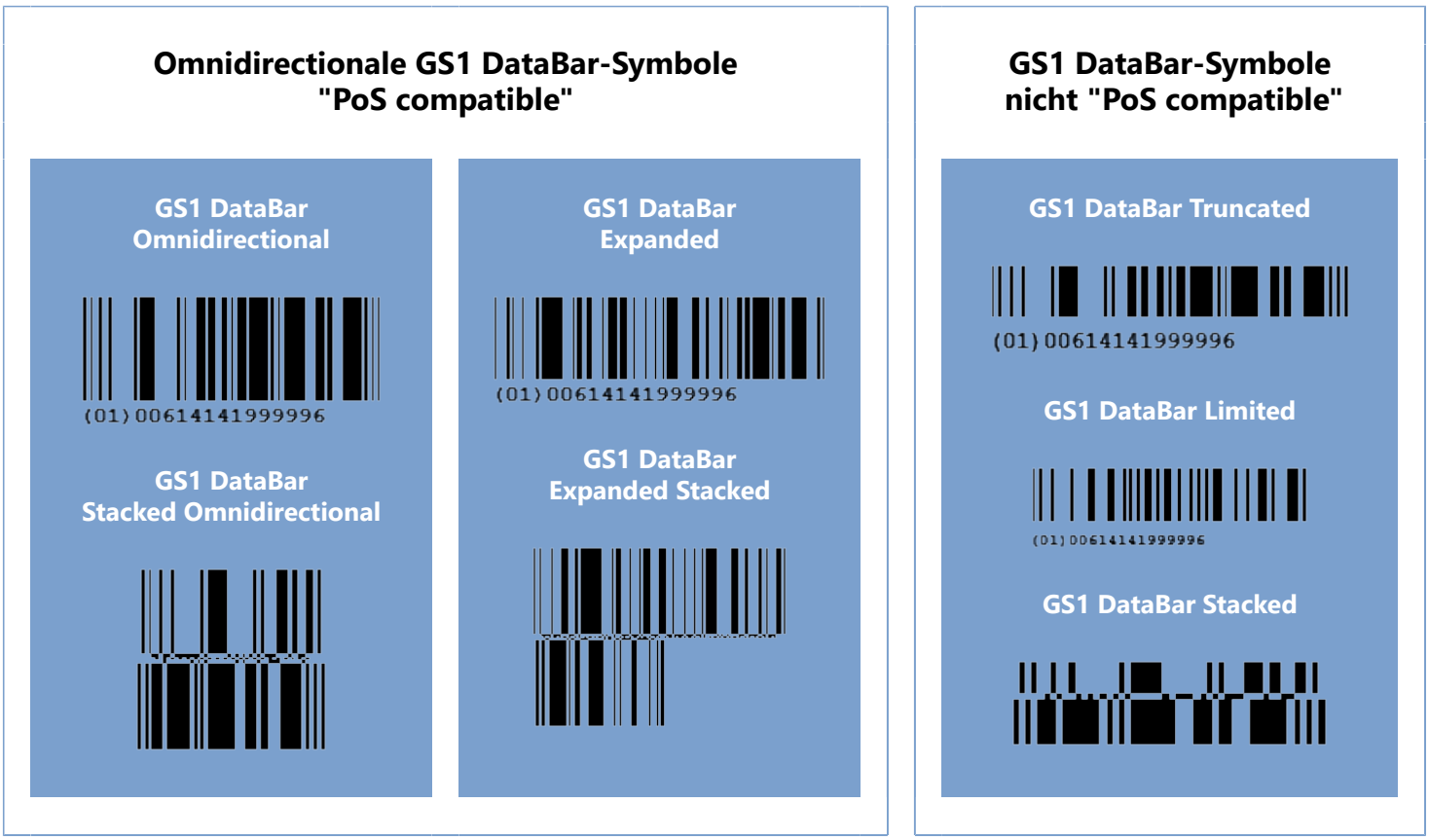
All **GS1 DataBar** barcodes encode a GTIN-12 or GTIN-13 in a 14-digit data structure. In order to make the GTIN-12 or GTIN-13 a 14-digit data structure, a leading zero or zeros is filled to the left of the GTIN. **GS1 DataBar Omnidirectional**, **GS1 DataBar Stacked Omnidirectional**, **GS1 DataBar Expanded**, and **GS1 DataBar Expanded Stacked** have omnidirectional scanning capability. **GS1 DataBar Truncated**, **GS1 DataBar Stacked** and **GS1 DataBar Limited** can only be scanned by a linear hand held or imaging scanning device: they cannot be scanned by omnidirectional scanners and are intended to be read by handheld scanners.

GS1 DataBar Stacked Omnidirectional is designed to condense the GTIN information into a more compact and square bar code suitable for use on smaller packages (such as the label stickers on fresh produce).

GS1 DataBar Limited, **GS1 DataBar Stacked** and **GS1 DataBar Truncated** are designed for very small item identification and are mainly used in the healthcare industry. Each encodes a GTIN-12 or GTIN-13 in 14-digit data structure. Only **GS1 DataBar Limited** uses an indicator digit 1.

In addition to encoding Application Identifier (01) GTIN, **GS1 DataBar Expanded** and **GS1 DataBar Expanded Stacked** can encode additional GS1 Application Identifiers such as sell-by date, weight, and lot number. Each symbol has a capacity of up to 74 characters. These attributes can help in controlling shrinkage, optimizing product replenishment, and improving the traceability of a product at the point of sale. They are seeing increased use in manufacturers' coupons.

This family of bar codes include:



See also

- GS1 DataBar Composite

GS1 DataBar Composite

The **GS1 Composite Symbology™** is the specification for a 2D barcode symbol included above a DataBar barcode to encode additional data.

The **GS1 Composite Symbology™** consists of a linear component, which encodes the items's primary data and an adjacent 2D composite component, which encodes supplementary data to the linear component such as e.g. expiration date, size, weight, lot or batch number.

The supplementary, 2D composite component (CC) will be one of the following:

- **CC-A** - up to 56 characters
- **CC-B** - up to 338 characters

Labelstar Office decides automatically, depending on the length of the composite data, which variant is used to display the barcode.

See also

- GS1 DataBar

GS1 DataMatrix

The **GS1 DataMatrix** is a 2D barcode with a high information density on relatively small space. A GTIN can be represented e.g. already on a space of 5 x 5 mm.

In the **GS1 DataMatrix** it is possible to code several data at the same time. It is mainly used in trade and industry particularly for labelling goods and pallets. It is usual to code additionally to the product code e.g. the weight and minimum durability date.

The **GS1 DataMatrix** is compatible to the existing GS1 standard and is protected for all GS1 applications.

Technical Specifications

Length	Variable
Valid characters	ASCII character set
Check digit	None

See also

- DataMatrix

GS1 Application Identifiers

Note

This list should only give an overview and has no claims to correctness and completeness.

AI	Data Content	Format (*)	FNC1 required (****)
00	Serial Shipping Container Code (SSCC)	n2 + n18 (17 data digits + 1 checksum digit)	
01	Global Trade Item Number (GTIN)	n2 + n14 (13 data digits + 1 checksum digit)	
02	GTIN of contained trade items	n2 + n14 (13 data digits + 1 checksum digit)	
10	Batch or lot number	n2 + an..20	(FNC1)
11 (**)	Production date (YYMMDD)	n2 + n6	
12 (**)	Due date (YYMMDD)	n2 + n6	
13 (**)	Packaging date (YYMMDD)	n2 + n6	
15 (**)	Best before date (YYMMDD)	n2 + n6	
16 (**)	Sell by date (YYMMDD)	n2 + n6	
17 (**)	Expiration date (YYMMDD)	n2 + n6	
20	Product variant	n2 + n2	
21	Serial number	n2 + an..20	(FNC1)
22	*Secondary data for specific health industry products (obsolete - this application identifier was retired in 2013)*	*n2 + an..29*	*(FNC1)*
240	Additional item identification	n3 + an..30	(FNC1)
241	Customer part number	n3 + an..30	(FNC1)
242	Made-to-Order variation number	n3 + n..6	(FNC1)
243	Packaging component number	n3 + an..20	(FNC1)
250	Secondary serial number	n3 + an..30	(FNC1)
251	Reference to source entity	n3 + an..30	(FNC1)
253	Global Document Type Identifier (GDTI)	n3 + n13 (12 data digits + 1 checksum digit) + an..17	(FNC1)
254	GLN extension component	n3 + an..20	(FNC1)
255	Global Coupon Number (GCN)	n3 + n13 (12 data digits + 1 checksum digit) + n..12	(FNC1)
30	Count of items (variable measure trade item)	n2 + n..8	(FNC1)
310 (***)	Net weight, kilograms (variable measure trade item)	n4 + n6	
311 (***)	Length or first dimension, metres (variable measure trade item)	n4 + n6	
312 (***)	Width, diameter, or second dimension, metres (variable measure trade item)	n4 + n6	
313 (***)	Depth, thickness, height, or third dimension, metres (variable measure trade item)	n4 + n6	
314 (***)	Area, square metres (variable measure trade item)	n4 + n6	
315 (***)	Net volume, litres (variable measure trade item)	n4 + n6	
316 (***)	Net volume, cubic metres (variable measure trade item)	n4 + n6	
320 (***)	Net weight, pounds (variable measure trade item)	n4 + n6	
321 (***)	Length or first dimension, inches (variable measure trade item)	n4 + n6	
322 (***)	Length or first dimension, feet (variable measure trade item)	n4 + n6	

323 (***)	Length or first dimension, yards (variable measure trade item)	n4 + n6	
324 (***)	Width, diameter, or second dimension, inches (variable measure trade item)	n4 + n6	
325 (***)	Width, diameter, or second dimension, feet (variable measure trade item)	n4 + n6	
326 (***)	Width, diameter, or second dimension, yards (variable measure trade item)	n4 + n6	
327 (***)	Depth, thickness, height, or third dimension, inches (variable measure trade item)	n4 + n6	
328 (***)	Depth, thickness, height, or third dimension, feet (variable measure trade item)	n4 + n6	
329 (***)	Depth, thickness, height, or third dimension, yards (variable measure trade item)	n4 + n6	
330 (***)	Logistic weight, kilograms	n4 + n6	
331 (***)	Length or first dimension, metres	n4 + n6	
332 (***)	Width, diameter, or second dimension, metres	n4 + n6	
333 (***)	Depth, thickness, height, or third dimension, metres	n4 + n6	
334 (***)	Area, square metres	n4 + n6	
335 (***)	Logistic volume, litres	n4 + n6	
336 (***)	Logistic volume, cubic metres	n4 + n6	
337 (***)	Kilograms per square metre	n4 + n6	
340 (***)	Logistic weight, pounds	n4 + n6	
341 (***)	Length or first dimension, inches	n4 + n6	
342 (***)	Length or first dimension, feet	n4 + n6	
343 (***)	Length or first dimension, yards	n4 + n6	
344 (***)	Width, diameter, or second dimension, inches	n4 + n6	
345 (***)	Width, diameter, or second dimension, feet	n4 + n6	
346 (***)	Width, diameter, or second dimension, yards	n4 + n6	
347 (***)	Depth, thickness, height, or third dimension, inches	n4 + n6	
348 (***)	Depth, thickness, height, or third dimension, feet	n4 + n6	
349 (***)	Depth, thickness, height, or third dimension, yards	n4 + n6	
350 (***)	Area, square inches (variable measure trade item)	n4 + n6	
351 (***)	Area, square feet (variable measure trade item)	n4 + n6	
352 (***)	Area, square yards (variable measure trade item)	n4 + n6	
353 (***)	Area, square inches	n4 + n6	
354 (***)	Area, square feet	n4 + n6	
355 (***)	Area, square yards	n4 + n6	
356 (***)	Net weight, troy ounces (variable measure trade item)	n4 + n6	
357 (***)	Net weight (or volume), ounces (variable measure trade item)	n4 + n6	
360 (***)	Net weight, quarts (variable measure trade item)	n4 + n6	
361 (***)	Net weight, gallons U.S. (variable measure trade item)	n4 + n6	
362 (***)	Net weight, quarts	n4 + n6	
363 (***)	Net weight, gallons U.S.	n4 + n6	
364 (***)	Net volumne, cubic inches (variable measure trade item)	n4 + n6	
365 (***)	Net volumne, cubic feet (variable measure trade item)	n4 + n6	
366 (***)	Net volumne, cubic yards (variable measure trade item)	n4 + n6	
367 (***)	Logistic volumne, cubic inches	n4 + n6	
368 (***)	Logistic volumne, cubic feet	n4 + n6	
369 (***)	Logistic volumne, cubic yards	n4 + n6	
37	Count of trade items	n2 + n..8	(FNC1)

390 (***)	Applicable amount payable or coupon value, local currency	n4 + n..15	(FNC1)
391 (***)	Applicable amount payable with ISO currency code	n4 + n3 + n..15	(FNC1)
392 (***)	Applicable amount payable, single monetary area (variable measure trade item)	n4 + n..15	(FNC1)
393 (***)	Applicable amount payable with ISO currency code (variable measure trade item)	n4 + n3 + n..15	(FNC1)
394 (***)	Percentage discount of a coupon	n4 + n4	(FNC1)
400	Customer's pruchase order number	n3 + an..30	(FNC1)
401	Global Identification Number for Consignment (GINC)	n3 + an..30	(FNC1)
402	Global Shipment Identification Number (GSIN)	n3 + n17 (16 data digits + 1 checksum digit)	(FNC1)
403	Routing code	n3 + an..30	(FNC1)
410	Ship to - Deliver to Global Location Number	n3 + n13 (12 data digits + 1 checksum digit)	
411	Bill to - Invoice to Global Location Number	n3 + n13 (12 data digits + 1 checksum digit)	
412	Purchased from Global Location Number	n3 + n13 (12 data digits + 1 checksum digit)	
413	Ship for - Deliver for - Forward to Global Location Number	n3 + n13 (12 data digits + 1 checksum digit)	
414	Identification of physical location - Global Location Number	n3 + n13 (12 data digits + 1 checksum digit)	
415	Global Location Number of the invoicing party	n3 + n13 (12 data digits + 1 checksum digit)	
416	GLN of the production or service location	n3 + n13 (12 data digits + 1 checksum digit)	
420	Ship to - Deliver to postal code within a single postal authority	n3 + an..20	(FNC1)
421	Ship to - Deliver to postal code with ISO country code	n3 + n3 + an..20	(FNC1)
422	Country of origin of a trade item (ISO country code)	n3 + n3	(FNC1)
423	Country of initial processing (ISO country code)	n3 + n3 + n..12	(FNC1)
424	Country of processing (ISO country code)	n3 + n3	(FNC1)
425	Country of disassembly (ISO country code)	n3 + n3 + n..12	(FNC1)
426	Country covering full process chain (ISO country code)	n3 + n3	(FNC1)
427	Country subdivision of origin	n3 + an..3	(FNC1)
7001	NATO Stock Number (NSN)	n4 + n13	(FNC1)
7002	UN/ECE meat carcasses and cuts classification	n4 + an..30	(FNC1)
7003	Expiration date and time (YYMMDDhhmm)	n4 + n10	(FNC1)
7004	Active potency	n4 + n..4	(FNC1)
7005	Catch area (http://www.fao.org/fishery/area/search/en)	n4 + an..12	(FNC1)
7006	First freeze date (YYMMDD)	n4 + n6	(FNC1)
7007	Harvest date (YYMMDDhhmmss)	n4 + n6 + n..6	(FNC1)
7008	Species for fishery purposes (http://www.fao.org/fishery/collection/asfis/en)	n4 + an..3	(FNC1)
7009	Fishing gear type (http://www.fao.org/fishery/cwp/handbook/M/en)	n4 + an..10	(FNC1)
7010	Production method	n4 + an..2	(FNC1)
7020	Refurbishment lot ID	n4 + an..20	(FNC1)
7021	Functional status	n4 + an..20	(FNC1)
7022	Revision status	n4 + an..20	(FNC1)
7023	Global Individual Asset Identifier (GIAI) on an assembly	n4 + an..30	(FNC1)

7030	Number of processor with ISO country code	n4 + n3 + an..27	(FNC1)
7031	Number of 1st processor with ISO country code	n4 + n3 + an..27	(FNC1)
7032	Number of 2nd processor with ISO county code	n4 + n3 + an..27	(FNC1)
7033	Number of 3rd processor with ISO country code	n4 + n3 + an..27	(FNC1)
7034	Number of 4th processor with ISO county code	n4 + n3 + an..27	(FNC1)
7035	Number of 5th processor with ISO country code	n4 + n3 + an..27	(FNC1)
7036	Number of 6th processor with ISO country code	n4 + n3 + an..27	(FNC1)
7037	Number of 7th processor with ISO country code	n4 + n3 + an..27	(FNC1)
7038	Number of 8th processor with ISO country code	n4 + n3 + an..27	(FNC1)
7039	Number of 9th processor with ISO country code	n4 + n3 + an..27	(FNC1)
710	National Healthcare Reimbursement Number (NHRN) - German PZN	n3 + an..20	(FNC1)
711	National Healthcare Reimbursement Number (NHRN) - France CIP	n3 + an..20	(FNC1)
712	National Healthcare Reimbursement Number (NHRN) - Spain CN	n3 + an..20	(FNC1)
713	National Healthcare Reimbursement Number (NHRN) - Brasil DRN	n3 + an..20	(FNC1)
8001	Roll products (width, length, core diameter, direction, splices)	n4 + n14	(FNC1)
8002	Cellular mobile telephone identifier	n4 + an..20	(FNC1)
8003	Global Returnable Asset Identifier (GRAI)	n4 + n14 (13 data digits + 1 checksum digit) + an..16	(FNC1)
8004	Global Individual Asset Identifier (GIAI)	n4 + an..30	(FNC1)
8005	Price per unit of measure	n4 + n6	(FNC1)
8006	Identification of components of a trade item	n4 + n14 + n2 + n2	(FNC1)
8007	International Bank Account Number (IBAN)	n4 + an..34	(FNC1)
8008	Date and time of production (YYMMDDhhmmss)	n4 + n8 + n..4	(FNC1)
8010	Component/Part Identifier (CPID)	n4 + an..30	(FNC1)
8011	Component/Part Identifier serial number (CPID SERIAL)	n4 + n..12	(FNC1)
8012	Software version	n4 + an..20	(FNC1)
8017	Global Service Relation Number to identify the relationship between an organisation offering services and the provider of services	n4 + n18	(FNC1)
8018	Global Service Relation Number to identify the relationship between an organisation offering services and the receipient of services	n4 + n18	(FNC1)
8019	Service Relation Instance Number (SRIN)	n4 + n..10	(FNC1)
8020	Payment slip reference number	n4 + an..25	(FNC1)
8100	*Coupon extended code - U.P.C. prefix and offer code (obsolete - not used anymore)*	*n4 + n6*	*(FNC1)*
8101	*Coupon extended code - U.P.C. prefix, offer code, and expiration date (obsolete - not used anymore)*	*n4 + n10*	*(FNC1)*
8102	*Coupon extended code - U.P.C. prefix (obsolete - not used anymore)*	*n4 + n2*	*(FNC1)*
8110	Coupon code identification for use in North America	n4 + an..70	(FNC1)
8111	Loyalty points of a coupon	n4 + n4	(FNC1)
8112	Paperless coupon code identification for use in North America	n4 + an..70	(FNC1)
8200	Extended Packaging URL	n4 + an..70	(FNC1)
90	Information mutually agreed between trading partners	n2 + an..30	(FNC1)

91 - 99	Company internal information	n2 + an..30	(FNC1)

(*) The first position indicates the length (number of digits) of the GS1 Application Identifier. The following value refers to the format of the data content. The following convention is applied:
n - numeric digit
an - any character
n3 - 3 numeric digits, fixed length
n..3 - up to 3 numeric digits
an..3 - up to 3 characters

(**) If only year and month are available, DD must be filled with two zeroes.

(***) The fourth digit of this GS1 Application Identifier indicates the implied decimal point position.
Example:
3100 - Net weight in kg without a decimal point
3102 - Net weight in kg with two decimal points

(****) All GS1 Application Identifiers indicated with (FNC1) are defined as of variable length and shall be delimited unless this element string is the last one to be encoded in the symbol.

Postal Barcodes

Postal barcodes allows mailers to use a single barcode to:

- Participate in multiple Postal Service programs simultaneously
- Track individual mailpieces
- Have greater visibility to the mailpiece through its journey in the mail stream

Supported Barcodes

- Deutsche Post Identcode
- Deutsche Post Leitcode
- USPS Intelligent Mail® Barcode
- USPS PostNet

Deutsche Post Identcode

The **Identcode** is a variant of Code 2 of 5 Interleaved, but with a different check digit. This code is used by the Deutsche Post AG (DHL) and serves the automatic distribution of freight parcels in the post-office centres.

Structure of the **Identcode**:

- **1..2:** Mail center (outgoing)
- **3..5:** Customer code
- **6..11:** Delivery number
- **12:** Check digit

Technical Specifications

Length	12
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- Deutsche Post Leitcode

Deutsche Post Leitcode

The **Leitcode** is a variant of Code 2 of 5 Interleaved, but with a different check digit. This code is used by the Deutsche Post AG (DHL) and serves the automatic distribution of freight parcels in the post-office centres.

Structure of the **Leitcode**:

- **1..5:** ZIP code
- **6..8:** Street's code number
- **9..11:** House number
- **12..13:** Product code
- **14:** Check digit

Technical Specifications

Length	14
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- Deutsche Post Identcode

USPS Intelligent Mail® Barcode

The **USPS Intelligent Mail® Barcode** is a 65-bar Postal Service™ barcode that is used to sort and track letters and flats. The **USPS Intelligent Mail® Barcode** was introduced nearly a decade ago, and now all mailers need to have it to receive Postal Service tracking and discounts. It applies to First-Class Mail® Letters and flats, Marketing Mail® Letters and flats, Periodical letters and flats, Bound Printed Matter flats, and Reply Mail.

The following information is encrypted in the **USPS Intelligent Mail® Barcode**:

- 20-digit Tracking Code, which contains the identification of the mailer plus unique mailpiece information
- 0, 5, 9 or 11-digit Routing Code, which consists of the Delivery Point Zip code

For more information about **USPS Intelligent Mail® Barcode**, see here.

Technical Specifications

Length	20-digit tracking code + 0, 5, 9 or 11-digit routing code
Valid characters	Digits 0-9
Check digit	Internal

See also

- USPS PostNet

USPS PostNet

PostNet (Postal Numeric Encoding Technique) is a barcode symbology used by the United States Postal Service to assist in directing mail. The ZIP Code or ZIP+4 code is encoded in half- and full-height bars. Most often, the delivery point is added, usually being the last two digits of the address or PO box number.

Technical Specifications

Length	5-12
Valid characters	Digits 0-9
Check digit	Modulo 10

See also

- USPS Intelligent Mail® Barcode

HIBC Barcodes

About the HIBC standards

In 1983 **HIBCC** was provided its initial mandate for the health care industry: develop a uniform barcode labeling standard for products shipped to hospitals. This relatively straight-forward concept represented a dramatic potential for the entire industry. By agreeing to place a consistent pattern of computer-readable barcodes on their products, manufacturers would provide a control mechanism that would yield enormous benefits to both their hospital customers and distributors.

The mandate to develop the standard grew out of a task force hosted by the American Hospital Association and composed of numerous other health care trade organizations, including those which ultimately founded HIBCC. Barcode technology had already proven a valuable tool for reducing labor costs and human error in other industries, such as retailing. In health care, the potential was even greater because of the impact errors can have on the quality of patient care.

The task force ultimately created the **Health Industry Barcode (HIBC)** Standard, composed of two parts: Part One, the HIBC Supplier Labeling Standard, is the basis for the Universal Product Number, and covers the formats used by suppliers of healthcare products. Part Two, the HIBC Provider Applications Standard covers the formats used for internal labeling by health care providers themselves.

The following year HIBCC was formed to administer the standard and issue the Labeler Identification Codes (LIC's) which identify individual manufacturers and are included within each barcode. The LIC database now additionally provides the identifiers which are key elements in EDI transaction message formats and the HIBCC Universal Product Number Repository (UPN®).

Barcode technology changes rapidly. Advances in hardware have made it possible for "scanners" to understand multiple symbologies including those which are used for small-package and unit-of-use labeling. These developments have in turn made barcode use far more useful and valuable to the industry.

HBIC data structure

- Unique data structure for short to long alphanumeric product codes
- Symbologies for marking large to smallest products
- ISO conforming specification
- Interoperability with any other coding system
- Suitable for any ISO Barcode, 2D-Barcode and RFID

To combine the Primary and Secondary Data use the HIBC LIC Concatenation Format. In this case, a slash (/) is used as a delimiter between the Primary and Secondary Data. One check character (Modulo 43) at the end of the symbol is used which will verify the entire symbol.

Example

The labels *HIBC.lbex* und *HIBC2.lbex* can be found in the installation folder in the *Samples\HIBC* directory or can be downloaded here.

Labelstar Office supports the ANSI/HIBC 2.6 Supplier Labeling Standard. For more information, see here.

Check Digit Calculations

A check digit is a form of redundancy check used for error detection. It consists of a single digit (sometimes more than one) computed by an algorithm from the other digits (or letters) in the sequence input. With a check digit, one can detect simple errors in the input of a series of characters (usually digits) such as a single mistyped digit or some permutations of two successive digits.

- Modulo 10 (EAN-13, GTIN-13)
- Modulo 10 (Code 2/5)
- Modulo 10 (Identcode/Leitcode)
- Modulo 10 (Luhn Algorithm)
- Modulo 11 (PZN-8)
- Modulo 11 (UPU - Universal Postal Union)
- Modulo 43

A complete list of all built-in check digit calculations can be found here.

Modulo 10 (EAN)

Modulo 10 (EAN) is used by many barcode symbologies, for example, EAN-13, GTIN-13.

Check digit calculation according to Modulo 10 with digits in an odd position (from right to left) being weighted 1 and even position digits being weighted 3.

Sample Calculation

Digits	4 0 1 2 3 4 5 9 8 7 6 5
Weights	1 3 1 3 1 3 1 3 1 3 1 3
Total of products	4 + 0 + 1 + 6 + 3 + 12 + 5 + 27 + 8 + 21 + 6 + 15 = 108
Remainder mod 10	108 mod 10 = 8 (108/10 = 10 remainder 8)
Subtract from 10	10 - 8 = 2
Check digit	**2**

See also

- Modulo 10 (Code 2/5)
- Modulo 10 (Identcode/Leitcode)
- Modulo 10 (Luhn Algorithm)

Modulo 10 (Code 2/5)

For **Code 2 of 5 barcodes** (e.g. Code 2 of 5 Interleaved), calculation of the check digit is done in accordance with Modulo 10 with a weighting of 3. For the calculation, it begins with the first digit from the left with the weight factor of 3. The individual products are added to get a sum. The difference between the product and the next full "tens" (rounding) gives the check digit.

Sample Calculation

Digits	1 2 3 4 5
Weights	3 1 3 1 3
Total of products	3 + 2 + 9 + 4 + 15 = 33
Remainder mod 10	33 mod 10 = 3 (33/10 = 3 remainder 3)
Subtract from 10	10 - 3 = 7
Check digit	**7**

See also

- Modulo 10 (EAN)
- Modulo 10 (Identcode/Leitcode)
- Modulo 10 (Luhn Algorithm)

Modulo 10 (Identcode/Leitcode)

For the Deutsche Post Identcode and the Deutsche Post Leitcode, calculation of the check digit is done in accordance with Modulo 10 with a weighting of 4 for odd positions and with a weighting of 9 for even positions.

Sample Calculation

Digits	0 1 2 3 4 5 6 7 8 9 0 1 2
Weights	4 9 4 9 4 9 4 9 4 9 4 9 4
Total of products	0 + 9 + 8 + 27 + 16 + 45 + 24 + 63 + 32 + 81 + 0 + 9 + 8 = 322
Remainder mod 10	322 mod 10 = 2 (322/10 = 32 remainder 2)
Subtract from 10	10 - 2 = 8
Check digit	**8**

See also

- Modulo 10 (EAN)
- Modulo 10 (Code 2/5)
- Modulo 10 (Luhn Algorithm)

Modulo 10 (Luhn Algorithm)

The **Luhn Algorithm** or **Luhn Formula**, also known as the "modulus 10" or "mod 10" algorithm, is a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers, IMEI numbers, National Provider Identifier numbers in US and Canadian Social Insurance Numbers. It was created in 1960 by IBM scientist Hans Peter Luhn.

The algorithm is in the public domain and is in wide use today. It is not intended to be a cryptographically secure hash function; it was designed to protect against accidental errors, not malicious attacks. Most credit cards and many government identification numbers use the algorithm as a simple method of distinguishing valid numbers from mistyped or otherwise incorrect numbers.

Sample Calculation

Digits	4 5 5 6 7 3 7 5 8 6 8 9 9 8 5
Weights	2 1 2 1 2 1 2 1 2 1 2 1 2 1 2
Multiply digits by weight	8 5 10 6 14 3 14 5 16 6 16 9 18 8 10
Total of product digits	8 + 5 + 1 + 6 + 5 + 3 + 5 + 5 + 7 + 6 + 7 + 9 + 9 + 8 + 1 = 85
Remainder mod 10	85 mod 10 = 5 (85/10 = 8 remainder 5)
Check digit	**5**

See also

- Modulo 10 (EAN)
- Modulo 10 (Code 2/5)
- Modulo 10 (Identcode/Leitcode)

Modulo 11 (PZN-8)

A check digit **Modulo 11** is e.g. used by PZN barcodes.

To calculate the check digit multiply all digits with a weighting factor. The weighting factor for PZN-8 is the position of the digit (multiply the first digit by 1, the second digit by 2, the third digit by 3 and so on till the seventh digit by 7) and for PZN-7 the position of the digit + 1 (multiply the first digit by 2, the second digit by 3, the third digit by 4 and so on till the sixth digit by 7). Then add up all the results, divide the sum by 11 and take the remainder.

If the check digit happens to be 10 then the PZN barcode is treated as invalid and is not released.

Sample Calculation PZN-8

Digits	3 6 3 1 9 4 2
Weights	1 2 3 4 5 6 7
Total of products	3 + 12 + 9 + 4 + 45 + 24 + 14 = 111
Remainder mod 11	111 mod 11 = 1 (111/11 = 10 remainder 1)
Check digit	**1**

Modulo 11 (UPU)

This check digit calulation is used for bar codes created according to the Universal Postal Union S10 bar code standard. The UPU S10 standard defines a system for assigning 13-character identifiers to items for the purpose of tracking and tracing them during shipping.

UPU S10 Structure

- **1..2:** two-letter service indicator code
- **3..10:** eight digit number
- **11:** check digit according to modulo 11, from 3..10
- **12..13:** two-letter ISO country code

Sample Calculation

Digits	`0 0 1 3 3 5 7 5`
Weights	`8 6 4 2 3 5 9 7`
Total of products	`0 + 0 + 4 + 6 + 9 + 25 + 63 + 35 = 142`
Remainder mod 11	`142 mod 11 = 10 (142/11 = 12 remainder 10)`
Subtract from 11	`11 - 10 = 1` If the result falls within the range 1 to 9, use the result as the check digit; if the result is 10, use 0 as the check digit; if the result is 11, use 5 as the check digit.
Check digit	**1**

Modulo 43

A check digit in accordance with Modulo 43 is e.g. used by Code 39 and the HIBC Barcodes.

First, all characters in the code are allocated reference numbers (see reference table). These reference numbers are added to get a sum. This sum is divided by 43. The remainder from this division (Modulo 43) equates to the check sum which is then represented consistent with the reference number by means of the reference table.

Sample Calculation

Digits	1 5 9 A Z
Reference numbers	1 5 9 10 35
Total of reference numbers	1 + 5 + 9 + 10 + 35 = 60
Remainder mod 43	60 mod 43 = 17 (60/43 = 1 remainder 17)
Check digit	**H**

Reference Table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
W	X	Y	Z	-	.	Space	$	/	+	%					
32	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**					

Error Correction

What is error correction?

2D barcodes can store very large amounts of information. Due to the high data density, some of the data can be used for error correction. Various methods reserve up to 30% of the data for error correction, so that readers can also read dirty or damaged barcodes. This enables very low error rates to be achieved, even under unfavorable circumstances.

Offsetting the benefit of high error correction levels is the fact that each increased level of error correction also increases the size of the printed barcode, resulting in bigger barcodes as the correction level is raised.

See also

- DataMatrix
- PDF417
- QR Code

GTIN - Global Trade Item Number

Global Trade Item Number (GTIN) is an identifier for trade items. Such identifiers are used to look up product information in a database (often by inputting the number through a bar code scanner pointed at an actual product) which may belong to a retailer, manufacturer, collector, researcher, or other entity. The uniqueness and universality of the identifier is useful in establishing which product in one database corresponds to which product in another database, especially across organizational boundaries.

GTINs may be 8, 12, 13 or 14 digits long, and each of these 4 numbering structures are constructed in a similar fashion, combining Company Prefix, Item Reference and a calculated Check Digit (GTIN-14 adds another component - the Indicator Digit, which can be 1-8).

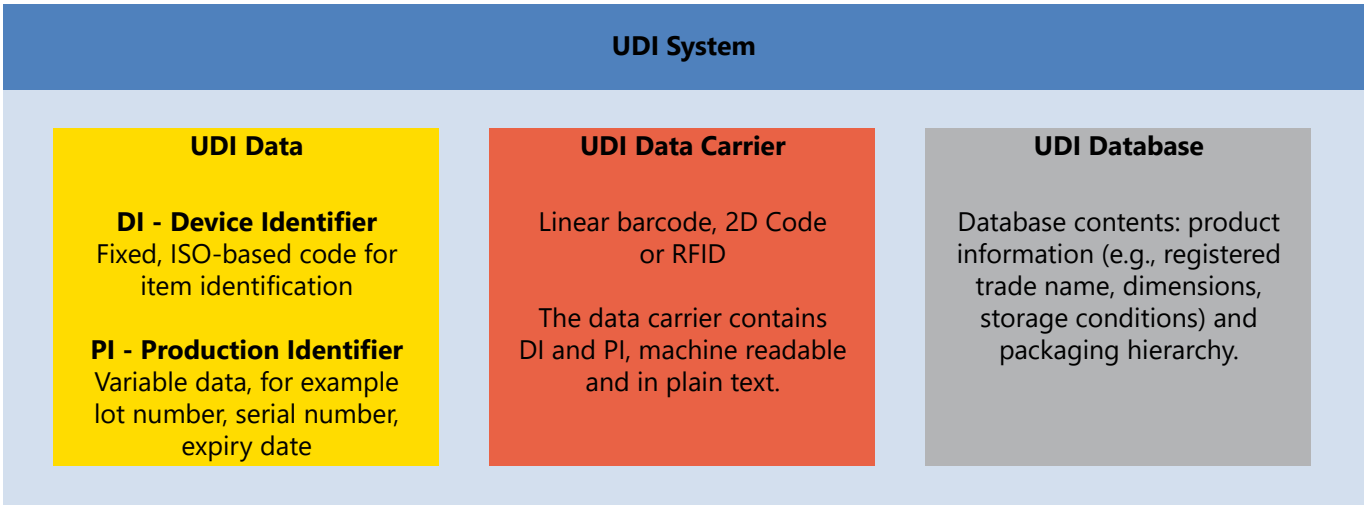
Name	Former Name
GTIN-8	EAN-8
GTIN-12	UPC-A
GTIN-13	EAN-13
GTIN 14	-

The 14-digit GTIN is formed by leading leading zeros from the previous 8, 12 and 13-digit article numbers:

GTIN-Typ	GTIN-Format													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
GTIN-8	0	0	0	0	0	0	N	N	N	N	N	N	N	C
GTIN-12	0	0	N	N	N	N	N	N	N	N	N	N	N	C
GTIN-13	0	N	N	N	N	N	N	N	N	N	N	N	N	C
GTIN 14	N	N	N	N	N	N	N	N	N	N	N	N	N	C

UDI - Unique Device Identification

Unique Device Identification (UDI) is a global normalized system for the identification of medical devices. The **UDI** should be visible, machine-readable and in plain text on the product itself. The **UDI** system also includes the development and operation of a **UDI** database (UDID) that contains a variety of information about the products.



A directive from the United States Food and Drug Administration (FDA) and the European Commission has introduced legislation regarding the **Unique Device Identification (UDI)** of medical devices and instruments. From September 2014, all medical class III devices and instruments that require direct part marking (DPM) must also carry a UDI. Class II and Class I devices will require UDI from September 2016 and September 2018 respectively, with the last deadline in September 2020.

See also

- HIBC Barcodes

ISO 3166 Country Codes

ISO 3166 is the International Standard for country codes and codes for their subdivisions. The purpose of ISO 3166 is to define internationally recognised codes of letters and/or numbers that can be be used to refer to countries and subdivisions.

Note

This list should only give an overview and has no claims to correctness and completeness.

Country	ISO Alpha-2 Code	ISO Alpha-3 Code	ISO Numeric Code
Afghanistan	AF	AFG	004
Ägypten	EG	EGY	818
Åland	AX	ALA	248
Albanien	AL	ALB	008
Algerien	DZ	DZA	012
Amerikanisch-Samoa	AS	ASM	016
Amerikanische Jungferninseln	VI	VIR	850
Andorra	AD	AND	020
Angola	AO	AGO	024
Anguilla	AI	AIA	660
Antarktika (Sonderstatus durch Antarktis-Vertrag)	AQ	ATA	010
Antigua und Barbuda	AG	ATG	028
Äquatorialguinea	GQ	GNQ	226
Argentinien	AR	ARG	032
Armenien	AM	ARM	051
Aruba	AW	ABW	533
Ascension	AC	ASC	
Aserbaidschan	AZ	AZE	031
Äthiopien	ET	ETH	231
Australien	AU	AUS	036
Bahamas	BS	BHS	044
Bahrain	BH	BHR	048
Bangladesch	BD	BGD	050
Barbados	BB	BRB	052
Belarus (Weißrussland)	BY	BLR	112
Belgien	BE	BEL	056
Belize	BZ	BLZ	084
Benin	BJ	BEN	204
Bermuda	BM	BMU	060
Bhutan	BT	BTN	064
Bolivien	BO	BOL	068
Bonaire, Sint Eustatius und Saba (Niederlande)	BQ	BES	535
Bosnien und Herzegowina	BA	BIH	070
Botswana	BW	BWA	072
Bouvetinseln	BV	BVT	074
Brazilien	BR	BRA	076
Britische Junfernsinseln	VG	VGB	092
Britisches Territorium im Indischen Ozean	IO	IOT	086
Brunei Darussalam	BN	BRN	096

Bulgarien	BG	BGR	100
Burkina Faso	BF	BFA	854
Burundi	BI	BDI	108
Chile	CL	CHL	152
China, Volksrepulik	CN	CHN	156
Cookinseln	CK	COK	184
Costa Rica	CR	CRI	188
Côte d'Ivoire (Elfenbeinküste)	CI	CIV	384
Curaçao	CW	CUW	531
Dänemark	DK	DNK	208
Deutschland	DE	DEU	276
Dominica	DM	DMA	212
Dominikanische Republik	DO	DOM	214
Dschibuti	DJ	DJI	262
Ecuador	EC	ECU	218
El Salvador	SV	SLV	222
Eritrea	ER	ERI	232
Estland	EE	EST	233
Falklandinseln	FK	FLK	238
Färöer	FO	FRO	234
Fiji	FJ	FJI	242
Finnland	FI	FIN	246
Frankreich	FR	FRA	250
Französisch-Guayana	GF	GUF	254
Französisch-Polynesien	PF	PYF	258
Französische Süd- und Antarktisgebiete	TF	ATF	260
Gabun	GA	GAB	266
Gambia	GM	GMB	270
Georgien	GE	GEO	268
Ghana	GH	GHA	288
Gibraltar	GI	GIB	292
Grenada	GD	GRD	308
Griechenland	GR	GRC	300
Grönland	GL	GRL	304
Guadeloupe	GP	GLP	312
Guam	GU	GUM	316
Guatemala	GT	GTM	320
Guernsey (Kanalinsel)	GG	GGY	831
Guinea	GN	GIN	324
Guinea-Bissau	GW	GNB	624
Guyana	GY	GUY	328
Haiti	HT	HTI	332
Heard- und McDonald-Inseln	HM	HMD	334
Hongkong	HK	HKG	344
Indien	IN	IND	356
Indonesien	ID	IDN	360
Irak	IQ	IRQ	368

Iran, Islamische Republik	IR	IRN	364
Irland	IE	IRL	372
Island	IS	ISL	352
Isle of Man	IM	IMN	833
Israel	IL	ISR	376
Italien	IT	ITA	380
Jamaika	JM	JAM	388
Japan	JP	JPN	392
Jemen	YE	YEM	887
Jersey (Kanalinsel)	JE	JEY	832
Jordanien	JO	JOR	400
Kaimaninseln	KY	CYM	136
Kambodscha	KH	KHM	116
Kamerun	CM	CMR	124
Kanada	CA	CAN	124
Kap Verde	CV	CVP	132
Kasachstan	KZ	KAZ	398
Katar	QA	QAT	634
Kenia	KE	KEN	404
Kirgisistan	KG	KGZ	417
Kiribati	KI	KIR	296
Kokosinseln	CC	CCK	166
Kolumbien	CO	COL	170
Komoren	KM	COM	174
Kongo, Demokratische Republik	CD	COD	180
Kongo, Republik	CG	COG	178
Korea, Demokratische Volksrepublik	KP	PRK	408
Korea, Republik	KR	KOR	410
Kroatien	HR	HRV	191
Kuba	CU	CUB	192
Kuwait	KW	KWT	414
Laos, Demokratische Volksrepublik	LA	LAO	418
Lesotho	LS	LSO	426
Lettland	LV	LVA	428
Libanon	LB	LBN	422
Liberia	LR	LBR	430
Libyen	LY	LBY	434
Liechtenstein	LI	LIE	438
Litauen	LT	LTU	440
Luxemburg	LU	LUX	442
Macau	MO	MAC	446
Madagaskar	MG	MDG	450
Malawi	MW	MWI	454
Malaysia	MY	MYS	458
Malediven	MV	MDV	462
Mali	ML	MLI	466
Malta	MT	MLT	470
Marokko	MA	MAR	504

Marshallinseln	MH	MHL	584
Martinique	MQ	MTQ	474
Mauritanien	MR	MRT	478
Mauritius	MU	MUS	480
Mayotte	YT	MYT	175
Mazedonien	MK	MKD	807
Mexiko	MX	MEX	484
Mikronesien	FM	FSM	583
Moldawien (Republik Moldau)	MD	MDA	498
Monaco	MC	MCO	492
Mongolei	MN	MNG	496
Montenegro	ME	MNE	499
Montserrat	MS	MSR	500
Mosambik	MZ	MOZ	508
Myanmar	MM	MMR	104
Namibia	NA	NAM	516
Nauru	NR	NRU	520
Nepal	NP	NPL	524
Neukaledonien	NC	NCL	540
Neuseeland	NZ	NZL	554
Niederlande	NL	NLD	528
Niederländische Antillen	AN	ANT	530
Niger	NI	NIC	558
Nigeria	NG	NGA	566
Niue	NU	NIU	558
Nördliche Marianen	MP	MNP	580
Norfolkinsel	NF	NFK	574
Norwegen	NO	NOR	578
Oman	OM	OMN	512
Österreich	AT	AUT	040
Pakistan	PK	PAK	586
Palästina	PS	PSE	275
Palau	PW	PLW	585
Panama	PA	PAN	591
Papua-Neuguinea	PG	PNG	598
Paraguay	PY	PRY	600
Peru	PE	PER	604
Philippinen	PH	PHL	608
Pitcairninseln	PN	PCN	612
Polen	PL	POL	616
Portugal	PT	PRT	620
Puerto Rico	PR	PRI	630
Qatar	QA	QAT	634
Réunion	RE	REU	638
Ruanda	RW	RWA	646
Rumänien	RO	ROU	642
Russische Föderation	RU	RUS	643
Salomonen	SB	SLB	090

Saint-Barthélemy	BL	BLM	652
Saint-Martin (franz. Teil)	MF	MAF	663
Sambia	ZM	ZMB	894
Samoa	WS	WSM	882
San Marino	SM	SMR	674
São Tomé und Príncipe	ST	STP	678
Saudi-Arabien	SA	SAU	682
Schweden	SE	SWE	752
Schweiz	CH	CHE	756
Senegal	SN	SEN	686
Serbien	RS	SRB	688
Seychelles	SC	SYC	690
Sierra Leone	SL	SLE	694
Simbabwe	ZW	ZWE	716
Singapur	SG	SGP	702
Sint Maarten (niederl. Teil)	SX	SXM	534
Slowakei	SK	SVK	703
Slowenien	SI	SVN	705
Somalia	SO	SOM	706
Spanien	ES	ESP	724
Sri Lanka	LK	LKA	144
St. Helena	SH	SHN	654
St. Kitts und Nevis	KN	KNA	659
St. Lucia	LC	LCA	662
Saint-Pierre und Miquelon	PM	SPM	666
St. Vincent und die Grenadinen	VC	VCT	670
Südafrika	ZA	ZAF	710
Sudan	SD	SDN	729
Südgeorgien und die Südlichen Sandwichinseln	GS	SGS	239
Südsudan	SS	SSD	728
Suriname	SR	SUR	740
Svalbard und Jan Mayen	SJ	SJM	744
Swasiland	SZ	SWZ	748
Syrien, Arabische Republik	SY	SYR	760
Tadschikistan	TJ	TJK	762
Taiwan, Republik China	TW	TWN	158
Tansania, Vereinigte Republik	TZ	TZA	834
Thailand	TH	THA	764
Timor-Leste	TL	TLS	626
Togo	TG	TGO	768
Tokelau	TK	TKL	772
Tonga	TO	TON	776
Trinidad und Tobago	TT	TTO	780
Tschad	TD	TCD	148
Tschechien	CZ	CZE	203
Tunesien	TN	TUN	788
Türkei	TR	TUR	792

Turkmenistan	TM	TKM	795
Turks- und Caicosinseln	TC	TCA	796
Tuvalu	TV	TUV	798
Uganda	UG	UGA	800
Ukraine	UA	UKR	804
Ungarn	HU	HUN	348
United States Minor Outlying Islands	UM	UMI	581
Uruguay	UY	URY	858
Usbekistan	UZ	UZB	860
Vanuatu	VU	VUT	548
Vatikanstadt	VA	VAT	336
Venezuela	VE	VEN	862
Vereinigte Arabische Emirate	AE	ARE	784
Vereinigte Staaten von Amerika	US	USA	840
Vereinigtes Königreich Großbritannien und Nordirland	GB	GBR	826
Vietnam	VN	VNM	704
Wallis und Futuna	WF	WLF	876
Weihnachtsinsel	CX	CXR	162
West-Sahara	EH	ESH	732
Yemen	YE	YEM	887
Zambia	ZM	ZMB	894
Zimbabwe	ZW	ZWE	716
Zypern	CY	CYP	196

ISO 4217 Currency Codes

The International Standardisation Organisation has introduced a coding system which assigns each currency a three-character currency code and a three-digit currency code. As some countries have experienced currency reforms, a country's currency code may change over time. For example, Mexico's currency code was changed from MXP to MXN when this country introduced the nuevo peso in January 1993.

Note

This list should only give an overview and has no claims to correctness and completeness.

Country	Currency Name	ISO Alpha-3 Code	ISO Numeric Code
Abu Dhabi	VAE Dirham	AED	784
Afghanistan	Afghani	AFN	971
Ägypten	Ägyptisches Pfund	EGP	818
Akrotiri	Euro	EUR	978
Albanien	Lek	ALL	008
Algerien	Algerischer Dinar	DZD	012
Amerikanische Jungferninseln	US-Dollar	USD	840
Amerikanisch-Samoa	US-Dollar	USD	840
Andorra	Euro	EUR	978
Angola	Kwanza	AOA	973
Anguilla	Ostkaribischer Dollar	XCD	951
Antigua und Barbuda	Ostkaribischer Dollar	XCD	951
Äquatorialguinea	CFA-Franc BEAC	XAF	950
Argentinien	Argentinischer Peso	ARS	032
Armenien	Dram	AMD	051
Aruba	Aruba-Florin	AWG	533
Aserbaidschan	Aserbaidschan-Manat	AZN	944
Äthiopien	Birr	ETB	230
Australien	Australischer Dollar	AUD	036
Bahamas	Bahama-Dollar	BSD	044
Bahrain	Bahrain-Dinar	BHD	048
Bangladesch	Taka	BDT	050
Barbados	Dollar	BBD	052
Belarus	Weißrussischer Rubel	BYR	974
Belgien	Euro	EUR	978
Belize	Belize-Dollar	BZD	084
Benin	CFA-Franc BCEAO	XOF	952
Bermuda	Bermuda-Dollar	BMD	060
Bhutan	Ngultrum	BTN	064
Bolivien	Boliviano	BOB	068
Bonaire	US-Dollar	USD	840
Bosnien und Herzegowina	Konvertible Mark	BAM	977
Botsuana	Pula	BWP	072
Brasilien	Brasilianischer Real	BRL	986
Britische Jungferninseln	US-Dollar	USD	840
Brunei Darussalam	Brunei-Dollar	BND	096
Bulgarien	Lew	BGN	975

Burkina Faso	CFA-Franc BCEAO	XOF	952
Burundi	Burundi-Franc	BIF	108
Cabo Verde	Kap-Verde-Escudo	CVE	132
Chile	Chilenischer Peso	CLP	152
China	Renminbi Yuan	CNY	156
Cookinseln	Neuseeland-Dollar	NZD	554
Costa Rica	Costa-Rica Colón	CRC	188
Côte d'Ivoire	CFA-Franc BCEAO	XOF	952
Curaçao	Antillen-Gulden	ANG	532
Dänemark	Dänische Krone	DKK	208
Deutschland	Euro	EUR	978
Dhekelia	Euro	EUR	978
Dominica	Ostkaribischer Dollar	XCD	951
Dominikanische Republik	Dominikanischer Peso	DOP	214
Dschibuti	Dschibuti-Franc	DJF	262
Dubai	VAE Dirham	AED	784
Ecuador	US-Dollar	USD	840
El Salvador	US-Dollar (Colón auslaufend)	USD	840
Eritrea	Nakfa	ERN	232
Estland	Euro	EUR	978
Falklandinseln	Falkland-Pfund	FKP	
Fidschi	Fidschi-Dollar	FJD	242
Finnland	Euro	EUR	978
Frankreich	Euro	EUR	978
Französisch-Guayana	Euro	EUR	978
Französisch-Polynesien	CFP-Franc	XPF	953
Gabun	CFA-Franc BEAC	XAF	950
Gambia	Dalasi	GMD	270
Gazastreifen	Neuer Schekel	ILS	376
Georgien	Lari	GEL	981
Ghana	Ghanaischer Cedi	GHS	936
Gibraltar	Gibraltar -Pfund	GIP	292
Grenada	Ostkaribischer Dollar	XCD	951
Griechenland	Euro	EUR	978
Grönland	Dänische Krone	DKK	208
Großbritannien	Pfund Sterling	GBP	826
Guadeloupe	Euro	EUR	978
Guam	US-Dollar	USD	840
Guatemala	Quetzal	GTQ	320
Guinea	Guinea-Franc	GNF	324
Guinea-Bissau	CFA-Franc BCEAO	XOF	952
Guyana	Guyana-Dollar	GYD	328
Haiti	Gourde	HTG	332
Honduras	Lempira	HNL	340
Hongkong	Hongkong-Dollar	HKD	344
Indien	Indische Rupie	HKD	356
Indonesien	Indonesische Rupiah	IDR	360

Irak	Irakischer Dinar	IQD	368
Iran	Iranischer Rial	IRR	364
Irland	Euro	EUR	978
Island	Isländische Krone	ISK	352
Israel	Neuer Schekel	ILS	376
Italien	Euro	EUR	978
Jamaika	Jamaikanischer Dollar	JMD	388
Japan	Yen	JPY	392
Jemen	Jemen-Rial	YER	886
Jordanien	Jordanischer Dinar	JOD	400
Kaimaninseln	Kaiman-Dollar	KYD	136
Kambodscha	Riel	KHR	116
Kamerun	CFA-Franc BEAC	XAF	950
Kanada	Kanadischer Dollar	CAD	124
Kasachstan	Tenge	KZT	398
Katar	Katar-Riyal	QAR	634
Kenia	Kenia-Schilling	KES	404
Kirgisistan	Som	KGS	417
Kiribati	Australischer Dollar	AUD	036
Kolumbien	Kolumbianischer Peso	COP	170
Komoren	Komoren-Franc	KMF	174
Kongo	CFA-Franc BEAC	XAF	950
Kongo, Demokratische Republik	Kongo-Franc	CDF	976
Korea, Demokratische Volksrepublik	Nordkoreanischer Won	KPW	408
Korea, Republik	Südkoreanischer Won	KRW	410
Kosovo	Euro	EUR	978
Kroatien	Kuna	HRK	191
Kuba	Kubanischer Peso	CUP	192
Kuba	Peso concertible	CUC	
Kuwait	Kuwait-Dinar	KWD	414
Laos	Kip	LAK	418
Lesotho	Loti	LSL	426
Lettland	Euro	EUR	978
Libanon	Libanesisches Pfund	LBP	422
Liberia	Liberianischer Dollar	LRD	430
Libyen	Libyscher Dinar	LYD	434
Liechtenstein	Schweizer Franken	CHF	756
Litauen	Euro	EUR	978
Luxemburg	Euro	EUR	978
Macau	Pataca	MOP	446
Madagaskar	Ariary	MAG	969
Malawi	Malawi-Kwacha	MWK	454
Malaysia	Ringgit	MYR	458
Malediven	Rufiyaa	MVR	462
Mali	CFA-Franc BCEAO	XOF	952
Malta	Euro	EUR	978

Marokko	Marokkanischer Dirham	MAD	504
Marshallinseln	US-Dollar	USD	840
Martinique	Euro	EUR	978
Mauretanien	Ouguiya	MRO	478
Mauritius	Mauritius-Rupie	MUR	480
Mayotte	Euro	EUR	978
Mazedonien	Mazedonischer Denar	MKD	807
Mexiko	Mexikanischer Peso	MXN	484
Midway	US-Dollar	USD	840
Mikronesien	US-Dollar	USD	840
Moldau	Moldawischer Leu	MDL	498
Monaco	Euro	EUR	978
Mongolei	Tögrög (auch Tugrig oder Tugrik)	MNT	496
Montenegro	Euro	EUR	978
Montserrat	Ostkaribischer Dollar	XCD	951
Mosambik	Metical	MZN	943
Myanmar	Kyat	MMK	104
Namibia	Namibischer Dollar	NAD	516
Nauru	Australischer Dollar	AUD	036
Nepal	Nepalesische Rupie	NPR	524
Neukaledonien	CFP-Franc	XPF	953
Neuseeland	Neuseeland-Dollar	NZD	554
Nicaragua	Córdoba Oro	NIO	558
Niederlande	Euro	EUR	978
Niger	CFA-Franc BCEAO	XOF	952
Nigeria	Naira	NGN	566
Niue	Neuseeland-Dollar	NZD	554
Nördliche Marianen	US-Dollar	USD	840
Nordzypern	Neue Lira	TRY	949
Norfolkinsel	Australischer Dollar	AUD	036
Norwegen	Norwegische Krone	NOK	578
Oman	Omanischer Rial	OMR	
Österreich	Euro	EUR	978
Pakistan	Pakistanische Rupie	PKR	586
Palau	US-Dollar	USD	840
Panama	Balboa	PAB	590
Papua-Neuguinea	Kina	PGK	598
Paraguay	Guarani	PYG	600
Peru	Nuevo Sol	PEN	604
Philippinen	Philippinischer Peso	PHP	608
Pitcairninseln	Neuseeland-Dollar	NZD	554
Polen	Zloty	PLN	985
Portugal	Euro	EUR	978
Puerto Rico	US-Dollar	USD	840
Réunion	Euro	EUR	978
Ruanda	Ruanda-Franc	RWF	646
Rumänien	Rumänischer Leu	RON	946
Russische Föderation	Russischer Rubel	RUB	643

Saba	US-Dollar	USD	840
Salomonen	Salomonen-Dollar	SBD	090
Sambia	Sambischer Kwacha	ZMK	894
Samoa	Tala	WST	882
San Marino	Euro	EUR	978
São Tomé und Príncipe	Dobra	STD	678
Saudi-Arabien	Saudi-Rial	SAR	682
Schweden	Schwedische Krone	SEK	752
Schweiz	Schweizer Franken	CHF	756
Senegal	CFA-Franc BCEAO	XOF	952
Serbien	Serbischer Dinar	RSD	941
Seychellen	Seychellen-Rupie	SCR	690
Sierra Leone	Leone	SLL	694
Simbabwe	Simbabwe-Dollar (ausgesetzt)	ZWD	716
Singapur	Singapur-Dollar	SGD	702
Sint Eustatius	US-Dollar	USD	840
Sint Maarten	Antillen-Gulden	ANG	532
Slowakei	Euro	EUR	978
Slowenien	Euro	EUR	978
Somalia	Somalia-Schilling	SOS	706
Spanien	Euro	EUR	978
Spitzbergen	Norwegische Krone	NOK	
Sri Lanka	Sri Lanka-Rupie	LKR	144
St. Barthélemy	Euro	EUR	978
St. Kitts und Nevis	Ostkaribischer Dollar	XCD	951
St. Lucia	Ostkaribischer Dollar	XCD	951
St. Martin	Euro	EUR	978
St. Pierre und Miquelon	Euro	EUR	978
St. Vincent und die Grenadinen	Ostkaribischer Dollar	XCD	951
Südafrika	Rand	ZAR	710
Sudan	Sudanesisches Pfund	SDG	938
Südgeorgien und die Südlichen Sandwichinseln	Pfund Sterling	GBP	
Südsudan	Südsudanesisches Pfund	SSP	938
Suriname	Suriname-Dollar	SRD	968
Swasiland	Lilangeni	SZL	748
Syrien	Syrisches Pfund	SYP	760
Tadschikistan	Somoni	TJS	972
Taiwan	Taiwan-Dollar	TWD	901
Tansania	Tansania-Schilling	TZS	834
Thailand	Baht	THB	764
Timor-Leste	US-Dollar (+ eigene Münzen)	USD	840
Togo	CFA-Franc BCEAO	XOF	952
Tokelau	Neuseeland-Dollar	NZD	
Tonga	Pa'anga	TOP	776
Trinidad und Tobago	Trinidad und Tobago Dollar	TTD	780
Tschad	CFA-Franc BEAC	XAF	950
Tschechische Republik	Tschechische Krone	CZK	203

Tunesien	Tunesischer Dinar	TND	788
Türkei	Neue Lira	TRY	949
Turkmenistan	Turkmenistan-Manat	TMT	795
Turks- und Caicosinseln	US-Dollar	USD	840
Tuvalu	Australischer Dollar	AUD	036
Uganda	Uganda-Schilling	UGX	800
Ukraine	Hrywna	UAH	980
Ungarn	Forint	HUF	348
Uruguay	Uruguayischer Peso	UYU	858
Usbekistan	So'm	UZS	860
Vanuatu	Vatu	VUV	548
Vatikanstadt	Euro	EUR	978
Venezuela	Bolívar fuerte	VEF	937
Vereinigte Arabische Emirate	VAE Dirham	AED	784
Vereinigte Staaten	US-Dollar	USD	840
Vietnam	Dong	VND	704
Wallis und Futuna	CFP-Franc	XPF	953
Weihnachtsinsel	Australischer Dollar	AUD	036
Westjordanland	Neuer Schekel	ILS	
Zentralafrikanische Republik	CFA-Franc BEAC	XAF	950
Zypern	Euro	EUR	978

Databases

A wealth of data which are outside the label can be used within the label. But how can I find and import these data in **Labelstar Office**? The answer is very simple: You have to create and use a data connection.

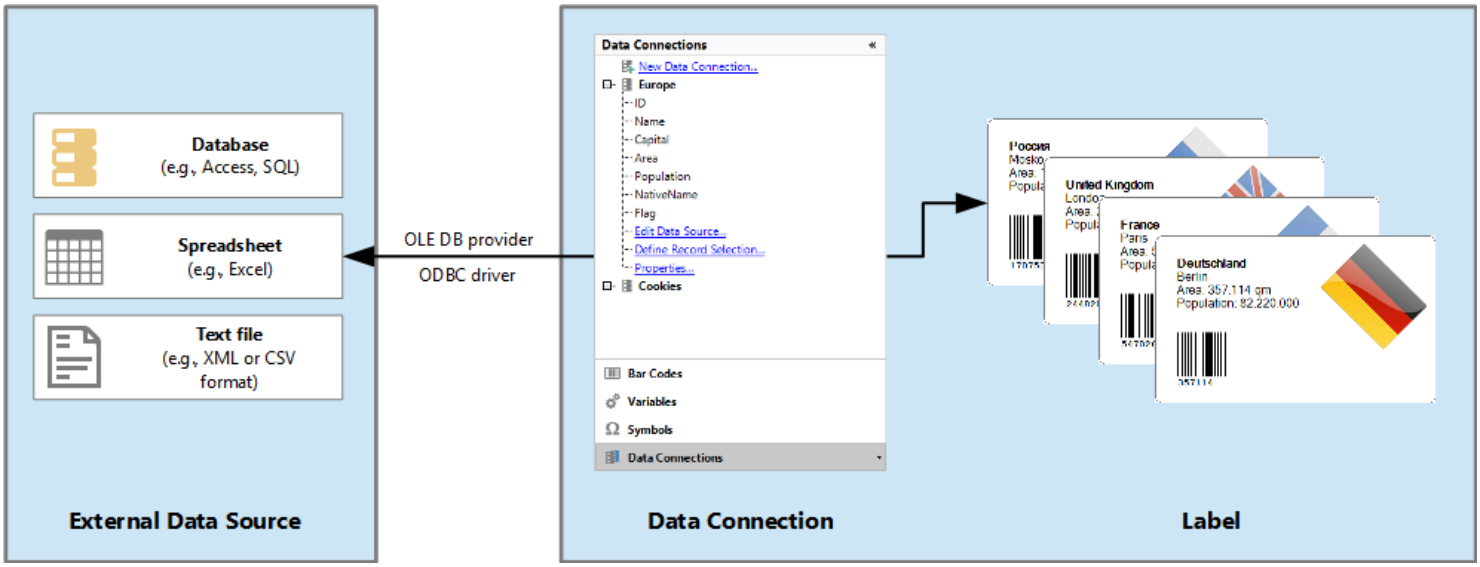
Data on a label can come from two different locations. The data may be stored directly within the label, or it may be stored in an external data source, such as a text file or a database. This external data source is connected to the label through a data connection, which is a set of information that describes how to locate, log in to, and access the external data source.

The main benefit of connecting to external data is that you can periodically analyze this data without repeatedly copying the data to your label, which is an operation that can be time consuming and prone to error.

To bring external data into **Labelstar Office**, you need access to the data. If the external data source that you want to access is not on your local computer, you may need to contact the administrator of the database for a password, user permissions, or other connection information. If the data source is a database, make sure that the database is not opened in exclusive mode. If the data source is a text file or a spreadsheet, make sure that another user does not have it open for exclusive access.

Many data sources also require an OLE DB provider or ODBC driver to coordinate the flow of data between **Labelstar Office**, the connection file, and the data source.

The following diagram summarizes the key points about data connections.



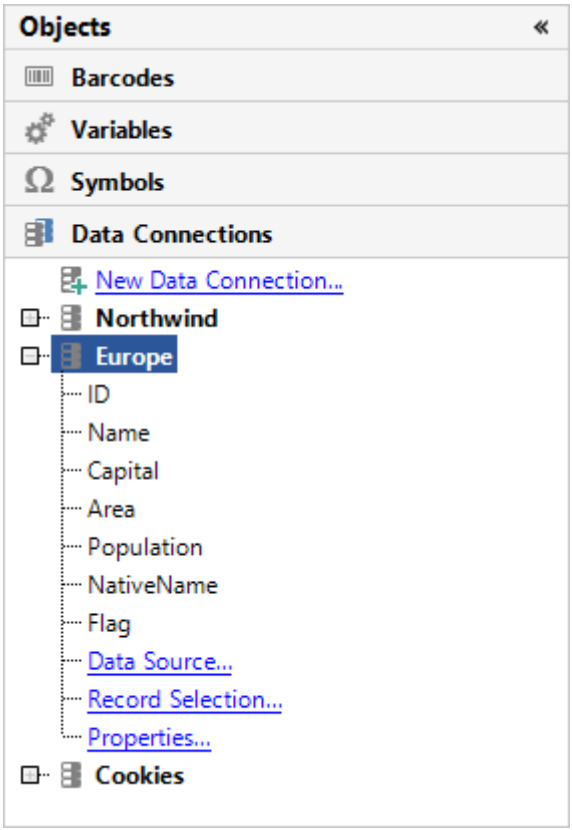
See also

- Create new data connection
- Refresh data connection
- Define logical connection path
- Create database label
- Share programs

Create New Data Connection

To specify a new data connection, proceed as follows:

1. Activate **Data Connections** view.
2. Click on **New Data Connection**.
   The **Data Connection Wizard** opens.
3. Select the data source you want to use.
4. Follow the instructions in the wizard.
5. After the successful definition, the new data connection is shown in the list and the associated database fields can be used on a label.



See also

- Refresh data connection

Refresh Data Connection

In **Labelstar Office** you can print labels with external data sources, e.g. SQL databases. To ensure fast data access **Labelstar Office** opens the data connections at program startup and loads the content into memory.

You can keep the information on your label up to date by "updating" the link to the data source. Every time the data connection is updated, the most current data is displayed, including all new and deleted data.

To refresh a data connection, proceed as follows:

1. Activate **Data Connections** view.
2. Select the data connection you want to update.
3. Right-click to open the context menu and click **Refresh**.

See also

- Create new data connection

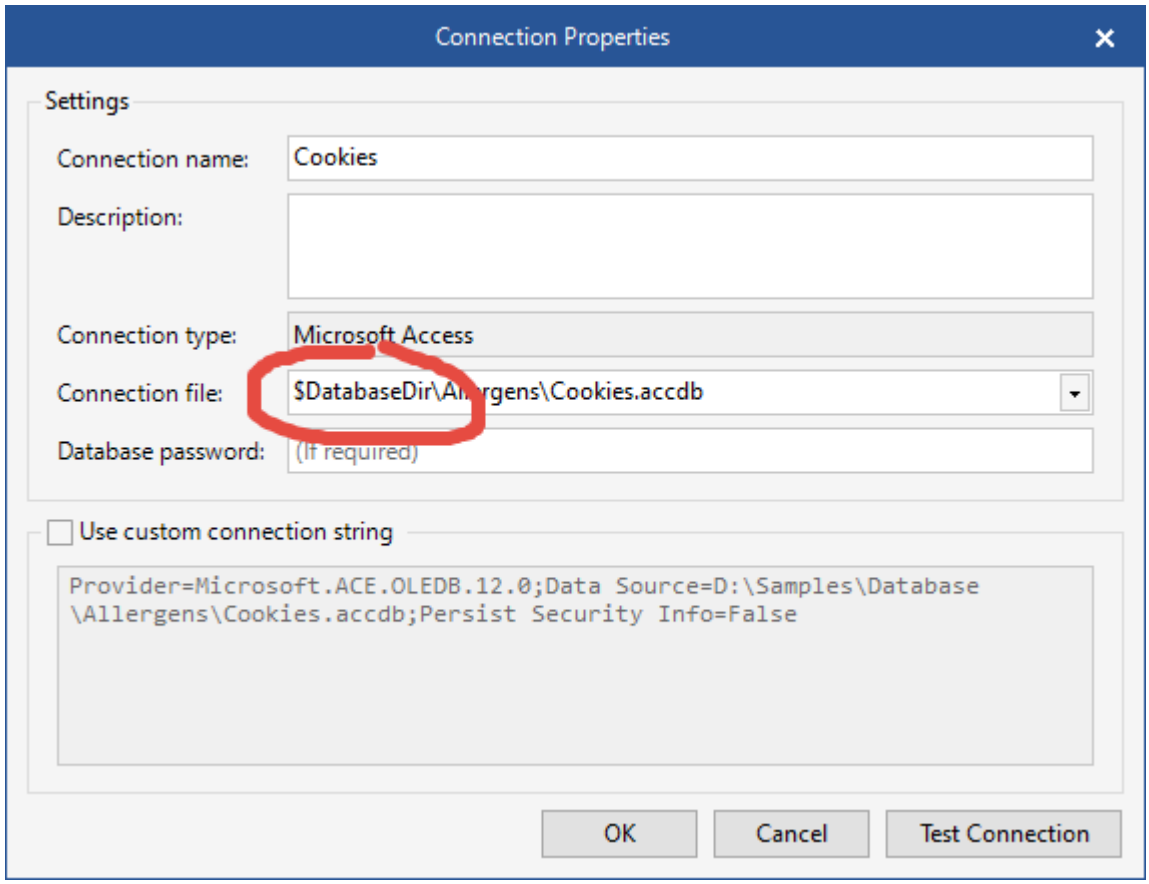
Define Logical Connection Path

You can use variables to define the file path of the data connection (for example, *$InstallDir\Samples\Allergens\Cookies.accdb*).

Labelstar Office supports a larger number of predefined variables for file management, but you can also create your own user-defined variables.

To define the file path of a connection file using a user-defined variable, proceed as follows:

1. Define a new user-defined variable (e.g., DatabaseDir = "D:\Samples\Database").
2. Activate **Data Connections** view and select a data connection.
3. Expand data connection and the click **Properties**.
4. Enter the *$DatabaseDir* variable in **Connection File** at the desired point as a placeholder.



OLE DB Provider and ODBC Driver

Labelstar Office uses the OLE DB interface in order to generate connections to external databases. Using OLE DB provider and ODBC driver, you can access a large number of common database systems. Alternatively, you can export the data from the database and save it in a file format (e.g. XML or CSV) which **Labelstar Office** is able to read directly.

Is the right OLE DB Provider or OLE Driver installed?

Normally, the operating system provides a range of OLE DB providers and ODBC drivers. In addition, you can buy OLE DB providers and ODBC drivers on the software market, download them online or get them directly from the producer of the database system. More information about installing these OLE DB providers or ODBC drivers can be found in the database documentation, or contact the database producer.

Note

Labelstar Office is a 32 bit application and can therefore only integrate 32 bit OLE DB providers and ODBC drivers.

SQL

What is SQL?

- **SQL** is a database language designed for the retrieval and management of data in relational databases
- **SQL** stands for "Structured Query Language"
- **SQL** is an ANSI (American National Standards Institute) standard

Why SQL?

- **SQL** allows users to access data in relational database management systems
- **SQL** allows users to describe the data
- **SQL** allows users to define the data in a database and manipulate that data
- **SQL** allows users to create views in a database
- ...

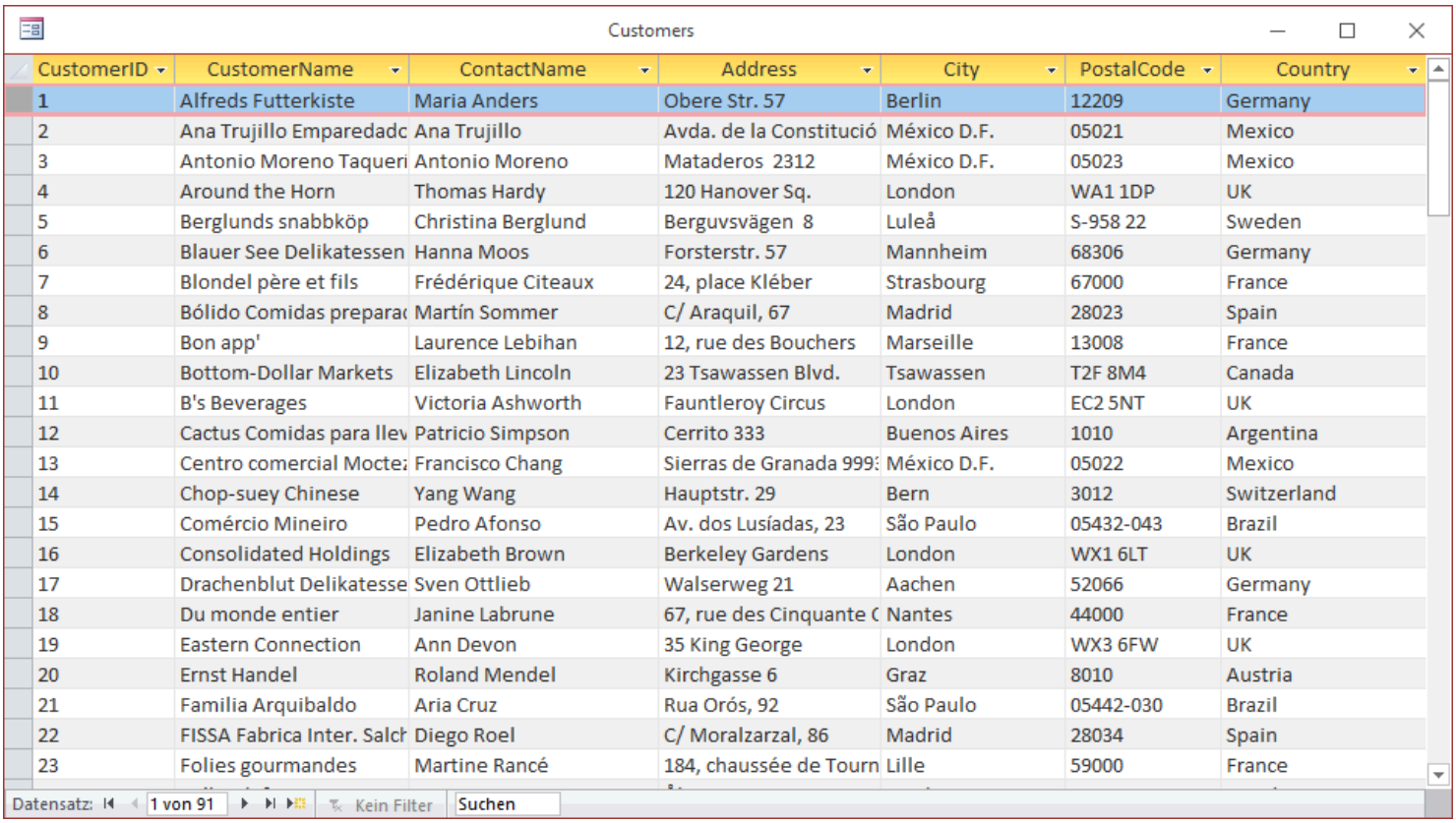
SQL is a Standard - BUT...

Although **SQL** is an ANSI standard, there are different versions of the SQL language. Most SQL database systems have their own proprietary extensions in addition to the standard SQL commands (such as SELECT, UPDATE, DELETE, INSERT, WHERE).

Relational Database Management Systems

RDBMS stands for "Relational Database Management System" and is the basic for **SQL**, and for all modern database systems such as MS SQL Server, IBM D2, Oracle, MySQL, and Microsoft Access.

The data in a relational database is stored in tables. A table is a collection of related data entries and consists of columns and rows. The rows of the tables are also called records.


Customers

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedadc	Ana Trujillo	Avda. de la Constitució	México D.F.	05021	Mexico
3	Antonio Moreno Taqueri	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
7	Blondel père et fils	Frédérique Citeaux	24, place Kléber	Strasbourg	67000	France
8	Bólido Comidas preparac	Martín Sommer	C/ Araquil, 67	Madrid	28023	Spain
9	Bon app'	Laurence Lebihan	12, rue des Bouchers	Marseille	13008	France
10	Bottom-Dollar Markets	Elizabeth Lincoln	23 Tsawassen Blvd.	Tsawassen	T2F 8M4	Canada
11	B's Beverages	Victoria Ashworth	Fauntleroy Circus	London	EC2 5NT	UK
12	Cactus Comidas para llev	Patricio Simpson	Cerrito 333	Buenos Aires	1010	Argentina
13	Centro comercial Moctez	Francisco Chang	Sierras de Granada 999:	México D.F.	05022	Mexico
14	Chop-suey Chinese	Yang Wang	Hauptstr. 29	Bern	3012	Switzerland
15	Comércio Mineiro	Pedro Afonso	Av. dos Lusíadas, 23	São Paulo	05432-043	Brazil
16	Consolidated Holdings	Elizabeth Brown	Berkeley Gardens	London	WX1 6LT	UK
17	Drachenblut Delikatesse	Sven Ottlieb	Walserweg 21	Aachen	52066	Germany
18	Du monde entier	Janine Labrune	67, rue des Cinquante (	Nantes	44000	France
19	Eastern Connection	Ann Devon	35 King George	London	WX3 6FW	UK
20	Ernst Handel	Roland Mendel	Kirchgasse 6	Graz	8010	Austria
21	Familia Arquibaldo	Aria Cruz	Rua Orós, 92	São Paulo	05442-030	Brazil
22	FISSA Fabrica Inter. Salch	Diego Roel	C/ Moralzarzal, 86	Madrid	28034	Spain
23	Folies gourmandes	Martine Rancé	184, chaussée de Tourn	Lille	59000	France

Datensatz: 1 von 91 Kein Filter Suchen

Every table is devided into smaller units called fields. The fields in the *Customers* table are *CustomerID*, *CustomerName*, *ContactName*, *Address*, *City*, *PostalCode*, and *Country*, and contain specific information about each record in the table.

A database can contain one or more tables. Each table is identified by a unique name (e.g., *Customers*).

See also

- Custom SQL statement
- Examples for SQL statements
- SQL Quick Reference

Custom SQL Statement

If you create a new data connection, **Labelstar Office** creates the associated SQL statement automatically according to the settings (table name, filter and sort citerias) you have made. If that is not enough, you can define a custom SQL statement.

To define a custom SQL statement, proceed as follows:

1. Activate **Data Connections** view.
2. Select a data connection and then click **Data Source**.
3. Click on **SQL Statement** tab and select **Use custom SQL statement**.
4. Enter SQL statement and click **OK** to save your changes.

More information about SQL can be found on the Internet (for example, enter the keyword "SQL tutorial" in your search engine).

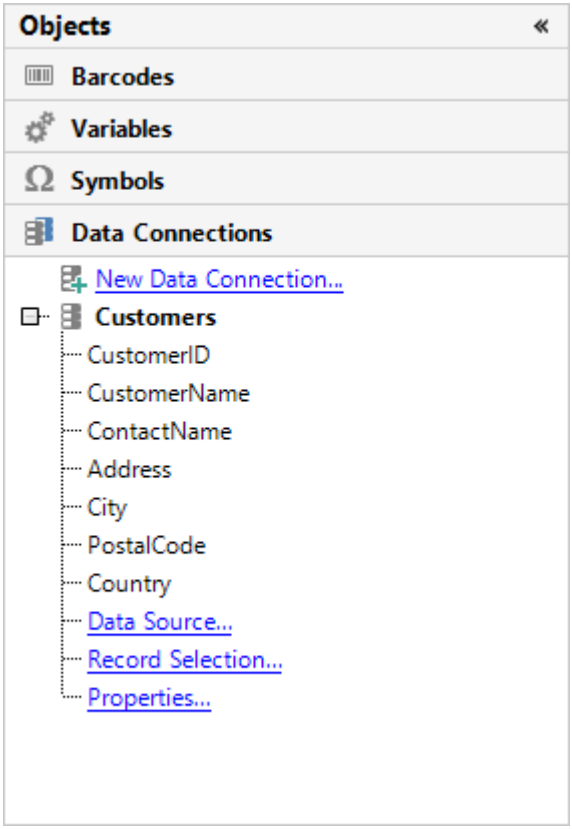
Examples for SQL Statements

Example

The **Sample.accdb** database used in the examples is based on the Northwind database and contains **Customers**, **Orders**, **Products**, **Shippers**, and **Suppliers** data tables.

The file can be found in the installation folder in the *Samples\SQL* directory or can be downloaded here.

In order to try out the examples directly, please create a new data connection to the sample database **Sample.accdb** (Microsoft Access, **Customers** table).



1. Click **Data Source** and select **SQL Statement** tab.
2. Select **Use custom SQL statement** and try one of the following examples:
   - **SELECT** - Retrieves certain records from one or more tables.
   - **WHERE** - Filters records.
   - **JOIN** - Combines rows from two ore more tables into a new virtual table.
   - **GROUP BY**- Groups the result-set by one ore more columns.

A quick reference to the most important SQL commands can be found here.

More information about SQL, and a complete list of all SQL commands can be found on the Internet (for example, enter the keyword "SQL tutorial" in your search engine).

SELECT - Retrieve Records

With the **SELECT** command you can retrieve certain records from one or more tables.

Syntax

```
SELECT column(s)
FROM table
```

```
SELECT *
FROM table
```

Examples

Select all columns from "Customers" table

```
SELECT * FROM Customers
```

Country	CustomerID	CustomerName	ContactName	Address	City
Germany	1	Alfreds Futterki...	Maria Anders	Obere Str. 57	Berlin
Mexico	2	Ana Trujillo Em...	Ana Trujillo	Avda. de la Con...	México D.F.
Mexico	3	Antonio Moren...	Antonio Moreno	Mataderos 2312	México D.F.
UK	4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London
Sweden	5	Berglunds snab...	Christina Bergl...	Berguvsvägen 8	Luleå
Germany	6	Blauer See Delik...	Hanna Moos	Forsterstr. 57	Mannheim
France	7	Blondel père et ...	Frédérique Cite...	24, place Kléber	Strasbourg

Select only the "CustomerName", "City" and "Country" columns from the "Customers" table

```
SELECT CustomerName, City, Country FROM Customers
```

Country	CustomerName	City
Germany	Alfreds Futterki...	Berlin
Mexico	Ana Trujillo Em...	México D.F.
Mexico	Antonio Moren...	México D.F.
UK	Around the Horn	London
Sweden	Berglunds snab...	Luleå
Germany	Blauer See Delik...	Mannheim
France	Blondel père et ...	Strasbourg

Rename columns

```
SELECT CustomerName AS Kunde, City AS Stadt, Country AS Land FROM Customers
```

Kunde	Stadt	Land
Alfreds Futterki...	Berlin	Germany
Ana Trujillo Em...	México D.F.	Mexico
Antonio Moren...	México D.F.	Mexico
Around the Horn	London	UK
Berglunds snab...	Luleå	Sweden
Blauer See Delik...	Mannheim	Germany
Blondel père et ...	Strasbourg	France

Select columns from different tables

```
SELECT CustomerName, ProductName FROM Customers, Products
```

CustomerName	ProductName
Alfreds Futterkiste	Chai
Alfreds Futterkiste	Chang
Alfreds Futterkiste	Aniseed Syrup
Alfreds Futterkiste	Chef Anton's Cajun Seasoning
Alfreds Futterkiste	Chef Anton's Gumbo Mix
Alfreds Futterkiste	Grandma's Boysenberry Spread
Alfreds Futterkiste	Uncle Bob's Organic Dried Pears

See also

- ORDER BY - Sort Records
- DISTINCT

ORDER BY - Sort Records

The **ORDER BY** command can be used to sort the selected records in ascending or descending order.

By default the records are sorted in ascending order. Use **DESC**, to sort the records in descending order.

Syntax

```
SELECT column(s)
FROM table
ORDER BY column(s) [ASC|DESC]
```

Examples

Sort records in ascending order

```
SELECT CustomerName, City, Country FROM Customers ORDER BY Country
```

CustomerName	City	Country
Rancho grande	Buenos Aires	Argentina
Océano Atlántico Ltda.	Buenos Aires	Argentina
Cactus Comidas para llevar	Buenos Aires	Argentina
Piccolo und mehr	Salzburg	Austria
Ernst Handel	Graz	Austria
Maison Dewey	Bruxelles	Belgium
Suprêmes délices	Charleroi	Belgium

Sort records in descending order

```
SELECT CustomerName, City, Country FROM Customers ORDER BY Country DESC
```

CustomerName	City	Country
LINO-Delicateses	I. de Margarita	Venezuela
GROSELLA-Restaurante	Caracas	Venezuela
HILARIÓN-Abastos	San Cristóbal	Venezuela
LILA-Supermercado	Barquisimeto	Venezuela
Great Lakes Food Market	Eugene	USA
Lazy K Kountry Store	Walla Walla	USA
Split Rail Beer & Ale	Lander	USA

Sort records across multiple columns

```
SELECT CustomerName, City, Country FROM Customers ORDER BY Country, CustomerName
```

CustomerName	City	Country
Cactus Comidas para llevar	Buenos Aires	Argentina
Océano Atlántico Ltda.	Buenos Aires	Argentina
Rancho grande	Buenos Aires	Argentina
Ernst Handel	Graz	Austria
Piccolo und mehr	Salzburg	Austria
Maison Dewey	Bruxelles	Belgium
Suprêmes délices	Charleroi	Belgium
Comércio Mineiro	São Paulo	Brazil
Familia Arquibaldo	São Paulo	Brazil

```
SELECT CustomerName, City, Country FROM Customers ORDER BY Country ASC, CustomerName DESC
```

CustomerName	City	Country
Rancho grande	Buenos Aires	Argentina
Océano Atlántico Ltda.	Buenos Aires	Argentina
Cactus Comidas para llevar	Buenos Aires	Argentina
Piccolo und mehr	Salzburg	Austria
Ernst Handel	Graz	Austria
Suprêmes délices	Charleroi	Belgium
Maison Dewey	Bruxelles	Belgium
Wellington Importadora	Resende	Brazil
Tradição Hipermercados	São Paulo	Brazil

DISTINCT

The **SELECT DISTINCT** command is used to eliminate duplicate values.

Syntax

```
SELECT DISTINCT column(s)
FROM table
```

Examples

```
SELECT Country FROM Customers ORDER BY Country
```

Country
Argentina
Argentina
Argentina
Austria
Austria
Belgium
Belgium
Brazil
Brazil

In this selection same countries are displayed serveral times in the list. If this is not desired, use **SELECT DESTINCT** instead. This will filter out all duplicate or multiple entries.

```
SELECT DISTINCT Country FROM Customers ORDER BY Country
```

Country
Argentina
Austria
Belgium
Brazil
Canada
Denmark
Finland
France
Germany

WHERE - Filter Records

The **WHERE** command is used to filter records. Only those records that fullfil a specified condition are displayed.

The **WHERE** command supports the following comparison operators:

- Equal (=) and not equal (<>)
- Greater than (>) and less than (<)
- Greater or equal (>=) and less or equal (<=)

Syntax

```
SELECT column(s)
FROM table
WHERE condition [AND|OR condition ...]
```

Examples

Get all customers who come from Germany

```
SELECT * FROM Customers WHERE Country = 'Germany'
```

Country	CustomerID	CustomerName	ContactName	Address	City	Posta
Germany	1	Alfreds Futterki...	Maria Anders	Obere Str. 57	Berlin	12209
Germany	6	Blauer See Delik...	Hanna Moos	Forsterstr. 57	Mannheim	68306
Germany	17	Drachenblut De...	Sven Ottlieb	Walserweg 21	Aachen	52066
Germany	25	Frankenversand	Peter Franken	Berliner Platz 43	München	80805
Germany	39	Königlich Essen	Philip Cramer	Maubelstr. 90	Brandenburg	14776
Germany	44	Lehmanns Mar...	Renate Messner	Magazinweg 7	Frankfurt a.M.	60528
Germany	52	Morgenstern G...	Alexander Feuer	Heerstr. 22	Leipzig	04179
Germany	56	Ottilies Käselad...	Henriette Pfalz...	Mehrheimerstr....	Köln	50739
Germany	63	QUICK-Stop	Horst Kloss	Taucherstraße 10	Cunewalde	01307
Germany	79	Toms Spezialitä...	Karin Josephs	Luisenstr. 48	Münster	44087
Germany	86	Die Wandernde...	Rita Müller	Adenauerallee ...	Stuttgart	70563

Get all orders whose order numbers are between 10250 und 10260

```
SELECT * FROM Orders
```

CustomerID	OrderDate	OrderID	ShipperID
65	04.08.1994	10248	1
68	05.08.1994	10249	2
58	08.08.1994	10250	2
9	08.08.1994	10251	3
73	09.08.1994	10252	3
85	10.08.1994	10253	1
79	11.08.1994	10254	3
34	12.08.1994	10255	2
84	15.08.1994	10256	1

```
SELECT * FROM Orders WHERE OrderID >= 10250 AND OrderID <= 10260
```

CustomerID	OrderDate	OrderID	ShipperID
58	08.08.1994	10250	2
9	08.08.1994	10251	3
73	09.08.1994	10252	3
85	10.08.1994	10253	1
79	11.08.1994	10254	3
34	12.08.1994	10255	2
84	15.08.1994	10256	1
76	16.08.1994	10257	2
34	17.08.1994	10258	3

See also

- LIKE Operator
- IN Operator
- BETWEEN Operator

LIKE Operator

The **LIKE** operator is used in **WHERE** to search for a specified pattern in a column. Two placeholders can be used in conjunction with the **LIKE** operator. The percent sign (%) represents zero, one, or multiple characters. The underscore (_) represents a single characters.

Syntax

```
SELECT column(s)
FROM table
WHERE column LIKE 'pattern'
```

Examples

Search for all customers whose name begins with an "a"

```
SELECT * FROM Customers WHERE CustomerName LIKE 'a%'
```

CustomerID	CustomerName	ContactName	Address	City
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Con...	México D.F.
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London

Search for all customers whose name ends with an "a"

```
SELECT * FROM Customers WHERE CustomerName LIKE '%a'
```

CustomerID	CustomerName	ContactName	Address	City
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.
13	Centro comercial Moctezuma	Francisco Chang	Sierras de Gran...	México D.F.
30	Godos Cocina Típica	José Pedro Freyre	C/ Romero, 33	Sevilla
61	Que Delícia	Bernardo Batista	Rua da Panifica...	Rio de Janeiro
62	Queen Cozinha	Lúcia Carvalho	Alameda dos C...	São Paulo
88	Wellington Importadora	Paula Parente	Rua do Mercad...	Resende
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki

Search for all customers whose name contains an "or"

```
SELECT * FROM Customers WHERE CustomerName LIKE '%or%'
```

CustomerID	CustomerName	ContactName	Address	City
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London
36	Hungry Coyote Import Store	Yoshi Latimer	City Center Plaz...	Elgin
40	La corne d'abondance	Daniel Tonini	67, avenue de l'...	Versailles
43	Lazy K Kountry Store	John Steel	12 Orchestra Te...	Walla Walla
52	Morgenstern Gesundkost	Alexander Feuer	Heerstr. 22	Leipzig
53	North/South	Simon Crowther	South House30...	London
55	Old World Delicatessen	Rene Phillips	2743 Bering St.	Anchorage
72	Seven Seas Imports	Hari Kumar	90 Wadhurst Rd.	London

Search for all customers whose name has an "r" in the second place

```
SELECT * FROM Customers WHERE CustomerName LIKE '_r%'
```

CustomerID	CustomerName	ContactName	Address	City
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London
17	Drachenblut Delikatessen	Sven Ottlieb	Walserweg 21	Aachen
20	Ernst Handel	Roland Mendel	Kirchgasse 6	Graz
25	Frankenversand	Peter Franken	Berliner Platz 43	München
26	France restauration	Carine Schmitt	54, rue Royale	Nantes
27	Franchi S.p.A.	Paolo Accorti	Via Monte Bian...	Torino
32	Great Lakes Food Market	Howard Snyder	2732 Baker Blvd.	Eugene
33	GROSELLA-Restaurante	Manuel Pereira	5ª Ave. Los Pal...	Caracas
60	Princesa Isabel Vinhos	Isabel de Castro	Estrada da saúd...	Lisboa

See also

- IN Operator
- BETWEEN Operator

IN Operator

The **IN** operator allows you to specify multiply values in a **WHERE** command.

Syntax

```
SELECT column(s)
FROM table
WHERE column IN (value1, value2, ...)
```

```
SELECT column(s)
FROM table
WHERE column IN (SELECT statement)
```

Examples

Search for all customers who come from Germany, France or Great Britain

```
SELECT * FROM Customers WHERE Country IN ('Germany', 'France', 'UK')
```

CustomerID	CustomerName	ContactName	Address	City
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim
7	Blondel père et fils	Frédérique Cite...	24, place Kléber	Strasbourg
9	Bon app'	Laurence Lebih...	12, rue des Bou...	Marseille
11	B's Beverages	Victoria Ashwor...	Fauntleroy Circus	London
16	Consolidated Holdings	Elizabeth Brown	Berkeley Garde...	London
17	Drachenblut Delikatessen	Sven Ottlieb	Walserweg 21	Aachen
18	Du monde entier	Janine Labrune	67, rue des Cin...	Nantes

It is equivalent to the following SQL statement:

```
SELECT * FROM Customers WHERE Country = 'Germany' OR Country = 'France' OR Country = 'UK'
```

As you can see, using the **IN** operator makes the statement easier to read and more efficient than using multiple **OR** operators.

Search for all customers who are not from Germany, France or Great Britain

```
SELECT * FROM Customers WHERE Country NOT IN ('Germany', 'France', 'UK')
```

CustomerID	CustomerName	ContactName	Address	City
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Con...	México D.F.
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.
5	Berglunds snabbköp	Christina Bergl...	Berguvsvägen 8	Luleå
8	Bólido Comidas preparadas	Martín Sommer	C/ Araquil, 67	Madrid
10	Bottom-Dollar Markets	Elizabeth Lincoln	23 Tsawassen Bl...	Tsawassen
12	Cactus Comidas para llevar	Patricio Simpson	Cerrito 333	Buenos Aires
13	Centro comercial Moctezuma	Francisco Chang	Sierras de Gran...	México D.F.
14	Chop-suey Chinese	Yang Wang	Hauptstr. 29	Bern
15	Comércio Mineiro	Pedro Afonso	Av. dos Lusíada...	São Paulo

Search for all customers who come from the same countries as the suppliers

```
SELECT * FROM Customers WHERE Country IN (SELECT Country FROM Suppliers)
```

CustomerID	CustomerName	ContactName	Address	City
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London
5	Berglunds snabbköp	Christina Bergl...	Berguvsvägen 8	Luleå
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim
7	Blondel père et fils	Frédérique Cite...	24, place Kléber	Strasbourg
8	Bólido Comidas preparadas	Martín Sommer	C/ Araquil, 67	Madrid
9	Bon app'	Laurence Lebih...	12, rue des Bou...	Marseille
10	Bottom-Dollar Markets	Elizabeth Lincoln	23 Tsawassen Bl...	Tsawassen
11	B's Beverages	Victoria Ashwor...	Fauntleroy Circus	London

See also

- LIKE Operator
- BETWEEN Operator

BETWEEN Operator

The **BETWEEN** operator selects values within a given range.

Syntax

```
SELECT column(s)
FROM table
WHERE column BETWEEN value1 AND value2
```

Examples

Search for all products whose price is between 10 and 20

```
SELECT * FROM Products WHERE Price BETWEEN 10 AND 20
```

ProductID	ProductName	SupplierID	CategoryID	Unit	Price
1	Chai	1	1	10 boxes x 20 b...	18
2	Chang	1	1	24 - 12 oz bottles	19
3	Aniseed Syrup	1	2	12 - 550 ml bott...	10
15	Genen Shouyu	6	2	24 - 250 ml bott...	15,5
16	Pavlova	7	3	32 - 500 g boxes	17,45
21	Sir Rodney's Sc...	8	3	24 pkgs. x 4 pie...	10
25	NuNuCa Nuß-...	11	3	20 - 450 g glasses	14
31	Gorgonzola Teli...	14	4	12 - 100 g pkgs	12,5
34	Sasquatch Ale	16	1	24 - 12 oz bottles	14

It is equivalent to the following SQL statement:

```
SELECT * FROM Products WHERE Price >= 10 AND Price <= 20
```

See also

- LIKE Operator
- IN Operator

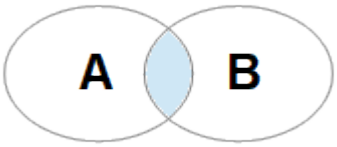
JOIN - Combine Tables

The **JOINs** commands are used to retreive data from multiple columns into a new virtual table.

There are four different types:

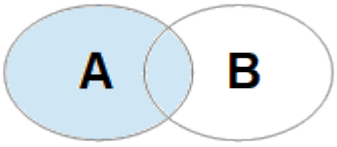
- INNER JOIN

  Returns all records that have matching values in both tables.



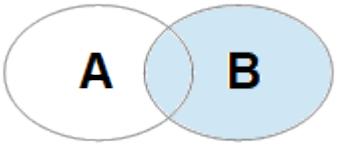
- LEFT JOIN

  Returns all records from the left table and the matching records from the right table.



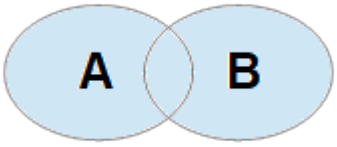
- RIGHT JOIN

  Returns all records from the right table and the matching records from the left table.



- FULL JOIN

  Returns all records that match either the left or the right table.

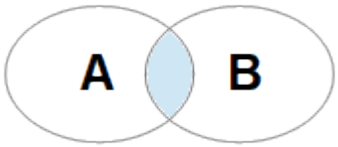


INNER JOIN

Returns all records that have matching values in both tables.

Syntax

```
SELECT column(s)
FROM tableA
INNER JOIN tableB
ON tableA.column = tableB.column
```



Examples

Extract from "Orders" table

Orders

CustomerID	OrderDate	OrderID	ShipperID
65	04.08.1994	10248	1
85	10.08.1994	10253	1
84	15.08.1994	10256	1
68	19.08.1994	10260	1
61	26.08.1994	10266	1
7	01.09.1994	10270	1
87	01.09.1994	10271	1
33	05.09.1994	10273	1

Datensatz: 1 von 830 Kein Filter Suchen

Extract from "Customers" table

Customers

CustomerID	CustomerName	ContactName	A
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Cc
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 23
4	Around the Horn	Thomas Hardy	120 Hanover S
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57
7	Blondel père et fils	Frédérique Citeaux	24, place Kléb
8	Bólido Comidas preparadas	Martín Sommer	C/ Araquil, 67

Datensatz: 1 von 91 Kein Filter Suchen

Get all orders with the corresponding customer information

```
SELECT Orders.OrderID, Customers.CustomerName FROM Orders INNER JOIN Customers ON
Orders.CustomerID = Customers.CustomerID ORDER BY Orders.OrderID
```

OrderID	CustomerName
10248	Rattlesnake Canyon Grocery
10249	Richter Supermarkt
10250	Pericles Comidas clásicas
10251	Bon app'
10252	Simons bistro
10253	Vins et alcools Chevalier
10254	Toms Spezialitäten
10255	Hanari Carnes
10256	Victuailles en stock

Get all orders with the corresponding customer and shipping information

```
SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName
FROM ((Orders INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID) INNER JOIN
Shippers ON Orders.ShipperID = Shippers.ShipperID)
ORDER BY Orders.OrderID
```

OrderID	CustomerName	ShipperName
10248	Rattlesnake Canyon Grocery	Speedy Express
10249	Richter Supermarkt	United Package
10250	Pericles Comidas clásicas	United Package
10251	Bon app'	Ferderal Shipping
10252	Simons bistro	Ferderal Shipping
10253	Vins et alcools Chevalier	Speedy Express
10254	Toms Spezialitäten	Ferderal Shipping
10255	Hanari Carnes	United Package
10256	Victuailles en stock	Speedy Express

See also

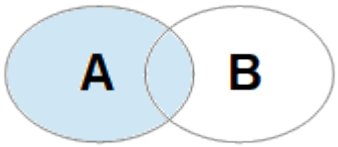
- LEFT JOIN
- RIGHT JOIN
- FULL JOIN

LEFT JOIN

Returns all records from the left table and the matching records from the right table.

Syntax

```
SELECT column(s)
FROM tableA
LEFT JOIN tableB
ON tableA.column = tableB.column
```



See also

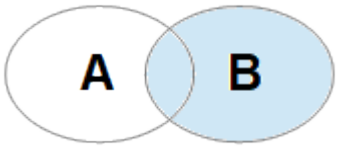
- INNER JOIN
- RIGHT JOIN
- FULL JOIN

RIGHT JOIN

Returns all records from the right table and the matching records from the left table.

Syntax

```
SELECT column(s)
FROM tableA
RIGHT JOIN tableB
ON tableA.column = tableB.column
```



See also

- INNER JOIN
- LEFT JOIN
- FULL JOIN

FULL JOIN

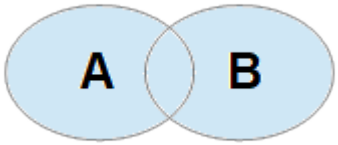
Returns all records that match either the left or the right table.

Note

This command is not supported by all database systems.

Syntax

```
SELECT column(s)
FROM tableA
FULL JOIN tableB
ON tableA.column = tableB.column
```



Examples

Get all customers and orders

```
SELECT Customers.CustomerName, Orders.OrderID FROM Customers FULL JOIN Orders ON
Customers.CustomerID = Orders.CustomerID
```

If your database system does not support the **FULL JOIN** command (MySQL and Mircosoft Access, for example, do not support **FULL JOIN**), you can use UNION ALL command to combine LEFT JOIN and RIGHT JOIN.

```
SELECT Customers.CustomerName, Orders.OrderID FROM Customers LEFT JOIN Orders ON
Customers.CustomerID = Orders.CustomerID UNION ALL
SELECT Customers.CustomerName, Orders.OrderID FROM Customers RIGHT JOIN Orders ON
Customers.CustomerID = Orders.CustomerID
```

CustomerName	OrderID
Alfreds Futterkiste	10648
Alfreds Futterkiste	10697
Alfreds Futterkiste	10707
Alfreds Futterkiste	10840
Alfreds Futterkiste	10957
Alfreds Futterkiste	11016
Ana Trujillo Emparedados y helados	10313
Ana Trujillo Emparedados y helados	10630
Ana Trujillo Emparedados y helados	10764

See also

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN

SELF JOIN

SELF JOIN is a regular join, but the table is joined with itself. The table is temporarily renamed into two reference tables.

Syntax

```
SELECT column(s)
FROM table A, table B
WHERE A.column = B.column
```

Examples

Get all customers who live in the same city

```
SELECT A.CustomerName AS CustomerName1, B.CustomerName AS CustomerName2, A.City FROM Customers A,
Customers B WHERE A.CustomerID <> B.CustomerID AND A.City = B.City ORDER BY A.City
```

CustomerName1	CustomerName2	City
Océano Atlántico Ltda.	Cactus Comidas para llevar	Buenos Aires
Cactus Comidas para llevar	Océano Atlántico Ltda.	Buenos Aires
Rancho grande	Océano Atlántico Ltda.	Buenos Aires
Rancho grande	Cactus Comidas para llevar	Buenos Aires
Océano Atlántico Ltda.	Rancho grande	Buenos Aires
Cactus Comidas para llevar	Rancho grande	Buenos Aires
Princesa Isabel Vinhos	Furia Bacalhau e Frutos do Mar	Lisboa
Furia Bacalhau e Frutos do Mar	Princesa Isabel Vinhos	Lisboa
B's Beverages	Around the Horn	London

See also

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN
- FULL JOIN

GROUP BY - Group Records

The **GROUP BY** command can be used to collect data across multiple records and group the result-set by one or more columns. The command is often used with aggregate functions (e.g., COUNT, MIN, MAX, ...).

Syntax

```
SELECT column(s)
FROM table
WHERE condition
GROUP BY column(s)
ORDER BY column(s)
```

Examples

Get the number of customers in each country, sorted from high to low

```
SELECT COUNT (CustomerID) AS [Count], Country FROM Customers GROUP BY Country ORDER BY COUNT
(CustomerID) DESC
```

Count	Country
13	USA
11	France
11	Germany
9	Brazil
7	UK
5	Mexico
5	Spain
4	Venezuela
3	Canada

SQL Quick Reference

The following table shows the most important SQL statements:

SQL Command	Syntax	Description
AND/OR	**SELECT** *column(s)* **FROM** *table* **WHERE** *condition* [**AND**|**OR** *condition* **...**]	Combine two or more conditions
AS (alias)	**SELECT** *column* **AS** *column_alias* **FROM** *table* or **SELECT** *column* **FROM** *table* **AS** *table_alias*	Assign a different name or alias to a column or a table (for an example see SELF JOIN)
BETWEEN	**SELECT** *column(s)* **FROM** *table* **WHERE** *column* **BETWEEN** *value1* **AND** *value2*	Retrieve records within a range (inclusive)
GROUP BY	**SELECT** *column(s)* **FROM** *table* **WHERE** *condition* **GROUP BY** *column(s)*	Group query result by one or more columns
HAVING	**SELECT** *column(s)* **FROM** *table* **WHERE** *condition* **GROUP BY** *column(s)* **HAVING** *condition*	Restrict the groups of returned records
IN	**SELECT** *column(s)* **FROM** *table* **WHERE** *column* **IN (***value1***,** *value2***, ...)**	Retrieve records matching any value in a value list (shorthand for multiple OR conditions)
INNER JOIN	**SELECT** *column(s)* **FROM** *tableA* **INNER JOIN** *tableB* **ON** *tableA.column* **=** *tableB.column*	Retrieve records that have matching values in both tables
LEFT JOIN	**SELECT** *column(s)* **FROM** *tableA* **LEFT JOIN** *tableB* **ON** *tableA.column* **=** *tableB.column*	Retrieve records from the left table and the matching records from the right table
RIGHT JOIN	**SELECT** *column(s)* **FROM** *tableA* **RIGHT JOIN** *tableB* **ON** *tableA.column* **=** *tableB.column*	Retrieve records from the right table and the matching records from the left table
FULL JOIN	**SELECT** *column(s)* **FROM** *tableA* **FULL JOIN** *tableB* **ON** *tableA.column* **=** *tableB.column*	Retrieve records that match either the left or the right table **Note:** This command is not supported by all database systems.
LIKE	**SELECT** *column(s)* **FROM** *table*	Retrieve records using wildcards in a WHERE command

	WHERE *column* LIKE *'pattern'*	
ORDER BY	**SELECT** *column(s)* **FROM** *table* **ORDER BY** *column(s)* [**ASC**|**DESC**]	Sort query results
SELECT	**SELECT** *column(s)* **FROM** *table*	Retrieve records from a table (specific columns)
SELECT *	**SELECT** * **FROM** *table*	Retrieve records from a table (all columns)
SELECT DISTINCT	**SELECT DISTINCT** *column(s)* **FROM** *table*	Retrieve records from a table (removes duplicates)
SELECT TOP	**SELECT TOP** *count*|*percent* *column(s)* **FROM** *table*	Retrieve records from a table and limit results **Note:** This command is not supported by all database systems.
UNION	**SELECT** *column(s)* **FROM** *tableA* **UNION** **SELECT** *column(s)* **FROM** *tableB*	Combine two result-sets (removes duplicates)
UNION ALL	**SELECT** *column(s)* **FROM** *tableA* **UNION ALL** **SELECT** *column(s)* **FROM** *tableB*	Combine two result-sets (includes duplicates)
WHERE	**SELECT** *column(s)* **FROM** *table* **WHERE** *condition* [**AND**|**OR** *condition* ...]	Filter results

More information about SQL, and a complete list of all SQL commands can be found on the Internet (for example, enter the keyword "SQL tutorial" in your search engine).

Sample

Example

The sample data can be found in the installation folder in the *Samples\Database* directory or can be downloaded from here.

Europe.accdb Database (Microsoft Access format)
Europe.csv Database (CSV format)
Europe.txt Database (Text format)
Europe.xml Database (XML format)
Import Europe_accdb data connection.lbdx Import file for data connection *Europe* (Microsoft Access format)
Import Europe_csv data connection.lbdx Import file for data connection *Europe* (CSV format)
Import Europe_txt data connection.lbdx Import file for data connection *Europe* (Text format)
Import Europe_xml data connection.lbdx Import file for data connection *Europe* (XML format)
Label.lbex Sample label

To print the sample label, proceed as follows:

1. Open **Labelstar Office**.
2. Import **Europe** data connection
   - Activate **data connection** view.
   - Right-click to open context menu and click **Import Data Connections**.
   - Navigate to samples folder and open **Import Europe_accdb data connection.lbdx**.
3. Open and print sample label

Program Options

In this dialog box, you can change several basic settings and customize the program to suit your personal preferences.

To change the program options, proceed as follows:

1. Select the **File** tab, and then click **Options**.
   The **Options** dialog box opens.
2. Change the desired settings.
3. Click **OK** to save your changes.

See also

- Application
  - General
  - Printing
  - Memory Card
  - Item Preferences
  - Standard Labels
- Labels
  - Preview
  - Logging
  - User Inputs
- General
  - Language
  - Check for Updates
  - File Management

General

Settings

Save label without prompt Activate this check box if the label should be saved without confirmation prompt.

Single document interface Activate this check box if only one label should be opened at a time. If this option is not selected then multiple labels can be opened in parallel.

At program startup

Select how **Labelstar Office** should behave at program startup:

- **Empty label** Opens a blank label.
- **Open recent label** Shows the recently opened label.
- **Open label** Opens a particular label. Click  to choose a file.
- **Show "Open File" dialog box** Displays the "Open File" dialog box to choose a label.

Printing

Print Settings

Default printer You can select the default printer for **Labelstar Office** here. **Labelstar Office** initially uses the Windows default printer, but you can select a different default printer for the print output. The Windows default printer and the **Labelstar Office** default printer are independent of one another. If you change one of the defaults, this does not affect the other one.

Note

The default printer which you select for **Labelstar Office** is a program setting, i.e. all labels which you print with **Labelstar Office** will be sent to this printer unless you select another printer for a specific label.

Configure Printer List By default, wherever you can select a printer in **Labelstar Office** (e.g. in the **Print** dialog box), all available Windows printers are listed. This list can be very long and contain printers that are not relevant for **Labelstar Office**. Click **Configure Printer List** to select which printers should be shown in the printer list and which not.

Quick Print

Number of copies Here, you can specify the number of copies to be printed by entering a number or changing the number with the help of the arrow control elements.

Memory Card

Memory Card Settings

Overwrite file on Memory Card without prompt Activate this check box if the label should be saved on the memory card without a confirmation prompt.

Save preview image as PNG file (recommended) Activate this check box if a preview image in PNG format should be saved on the memory card in addition to the label.

Save preview image as JPEG file Activate this check box if a preview image in JPEG format should be saved on the memory card in addition to the label.

Save Memory Card items first Activate this check box if the Memory Card items should be saved on the Memory Card before saving the label on the Memory Card.

Append file extension (.prn) Activate this check box if the .prn file name extension should be appended to the label names by default. If this option is not selected then the file path is created without a file name extension.

Folder Settings

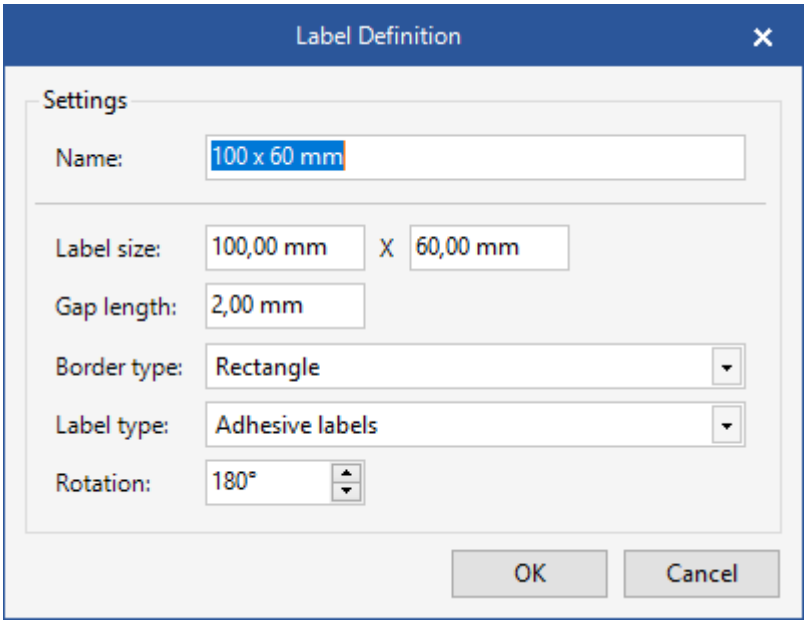
Default file location (printer-internal drive) Here, you can enter the default directory which should be used for creating the file path.

Item Preferences

This tabs allows you to set different default item settings. Any items (texts, barcodes, images, ...) you insert on the label after changing the default settings will receive the values you specified.

Standard Labels

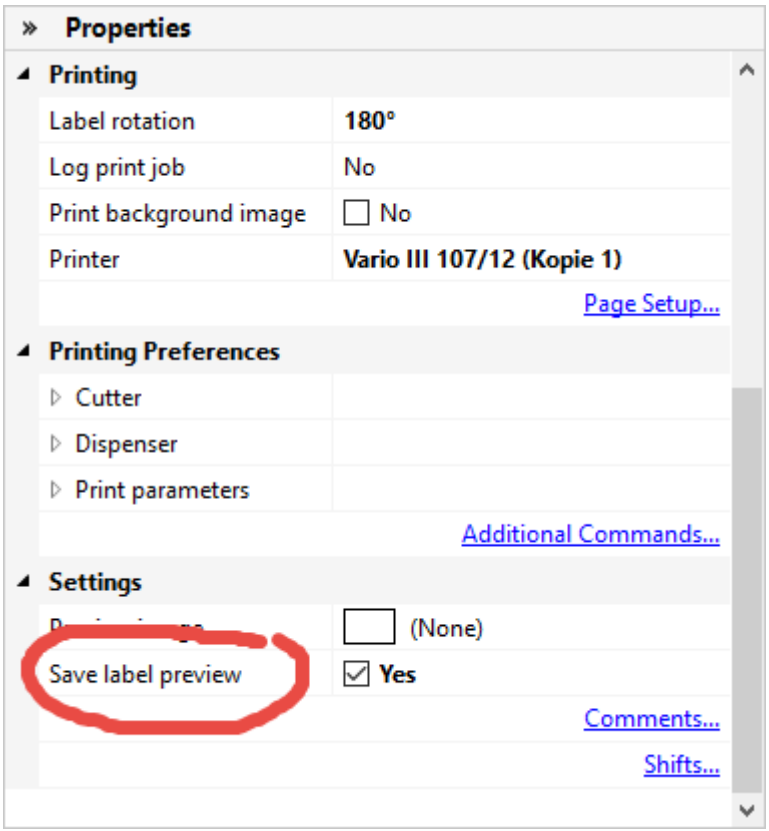
Use this tab to define your own standard labels. This label definitions can then be used for creating new labels. This results in the option of determining not only the label size but also other label settings.



Preview

In this tab, you can specify different settings (size, output format and color depth) of the preview image which is saved parallel to the label.

Activate the option **Always save preview images with labels** if you want to save a preview to each label. Do you want to activate the label preview only for particular labels then activate the option **Save label preview** in the label settings.



Logging

In this tab, you can change the settings for logging.

Log Settings

Activate the **Enable logging for all labels** option if you want every print job to be logged. For information about how to log only the print jobs of individual labels, see Activate and Deactivate Logging.

Log File Settings

You can specify where the log file is to be saved.

Location Enter the name of the folder in which the log file is to be created or click on  to browse the folder.

File name Enter a fixed file name or use placeholder *%date%*, *%time%*, *%labelname%*, *%printername%*, which are replaced by current values to define a variable file name (e.g. *%labelname%%date%.log*).

Note

Select only one path on which all users have access. Select never only *C:* or *C:\Windows*. If at all, please create a new folder where the program can save its log files (for example, *C:\Log*).

Max file size Specifies the maximum file size in MB. If the maximum size is reached, a new log file is created. Each new file name consists of the original file name with an ascending number.

Overwrite existing log file Activate this check box if the log file should overwrite and replace the existing log file.

Log File Format

In this section you can define the file format. The log file is saved in CSV format. For more information about data to be saved, see Logging.

User Inputs

Use Enter key as Tab key Activate this check box if the input focus should move to the next item in the activation sequence when RETURN is pressed. The dialog box is closed automatically (equivalent to clicking on **OK**) when the input focus has reached the last item in the tab order.

Show error message box Activate this check box if errors should be show with a notification in addition to a red border.

Group checkboxes Activate this check box if you want the check boxes to be grouped in the dialog box.

Make all checkboxes the same size Activate this option if the check boxes in the dialog box should be arranged in columns.

Language

In this tab, you can change the language of the menus and dialog boxes used in the **Labelstar Office** applications.

Note

The system language which is currently set will be used by default (provided that the language is available, if not then English will be used).

Note

If you changed the languge, you must quit and restart all applications that you have just opened.

Check for Updates

Here, you can specify if and how often **Labelstar Office** should search for program updates.

Note

In order to search for updates, your computer must have an internet connection and your firewall must not block access.

File Management

In this tab, you can change the location of the directories and files used in the program.

To change the location, proceed as follows:

1. Select an entry.
2. Click  .
   The dialog box **Select File/Select Folder** is opened.
3. Choose a new location.

Logging

With the logging you can retrace which data when, by whom and on which printer was printed.

Which information is saved at logging?

The logging option contained in **Labelstar Office** logs the following features:

- Date/Time of printing
- Number of copies
- Page name
- Label name
- Printer name
- User name
- Field contents

Define Settings for Log Files

In the **Options** dialog box on the Logging tab, you can specify the path and the format of the log files.

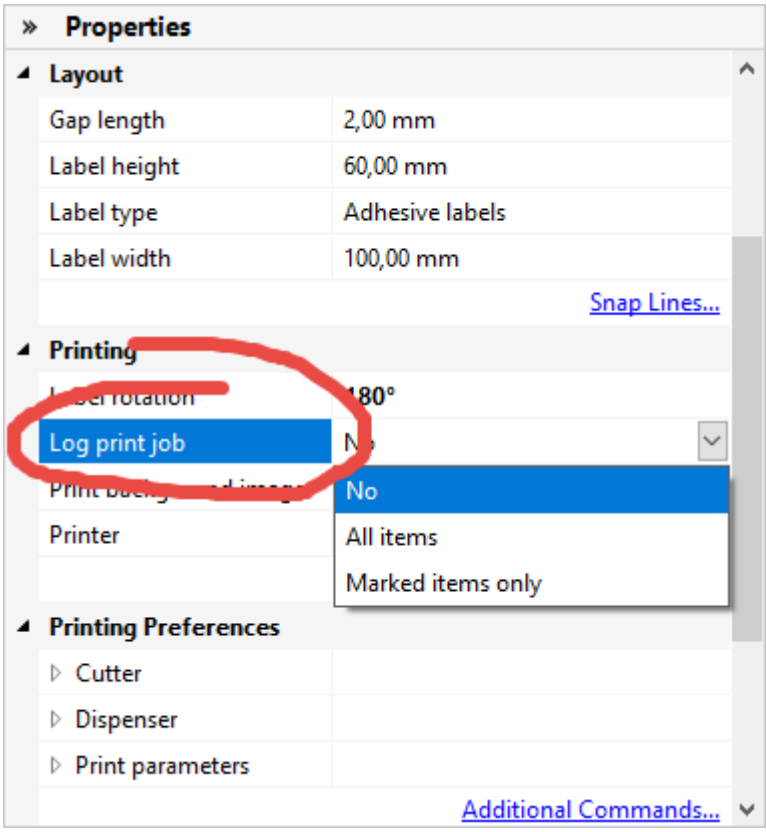
See also

- Activate/Deactivate logging

Activate and Deactivate Logging

In the **Options** dialog box, on the Logging tab, activate **Enable logging for all labels**, if you want every print job to be logged automatically. If you only want to log the print jobs of individual labels, please proceed as follows:

1. Select **Label Properties** and click **Log print job**.



2. To activate logging select one of the following options:
   - **All items** All item contents are logged.
   - **Marked items only** Only the contents of the items are logged, in which the **Log** option is enabled.
3. To deactivate logging select **No**.

See also

- Logging

Markup Tags

With the help of **markup tags** formatting instructions can be inserted into the text.

Note

Markup tags must be well formatted. That means that all tags must be properly closed and all attributes must have values enclosed in single quotes.

Note

Printer fonts and RTL fonts (e.g. Hebrew and Arabic) do not support **markup tags**.

The following table describes the formatting instructions.

Markup Tag	Description	Examples
b	Defines bold text.	bold text -> **bold text**
br	Defines a single line break.	line 1 line 2 -> line 1 line 2
em	Defines emphasized text (same as italic text).	emphasized text -> *emphasized text*
font	Defines font, color, and size for text. *Arributes:* • ***size**: Font size* • ***name**: Font name* • ***color**: Font color*	Example text -> Example text Example text -> Example **text**
i	Defines italic text.	italic text -> *italic text*
rtl	Right-to-left text.	<rtl>Example text</rtl> -> txet elpmaxE
shadow	Defines text with a shadow. *Attributes:* • ***color**: Shadow color (Default: Black)* • ***offset**: Shadow offset (Default: 1,1)* • ***strength**: Shadow size (Default: 1,1)* • ***style**: Shadow style (Default: Solid)* Solid Blurred	<shadow style='Blurred'>Example text</shadow> -> Example text <shadow color='Black'>Example text</shadow> -> Example text
strike	Defines strikethrough text.	<strike>strikethrough text</strike> -> ~~strikethrough text~~
stroke	Defines stroked text. *Attributes:* • ***width**: Border width (Default: 1)* • ***color**: Border color (Default: Black)*	<stroke color='#FF0000'>Example text</stroke> -> Example text
strong	Defines important text.	important text -> **important text**
sub	Defines subscripted text.	H₂O -> H_2O
sup	Defines superscripted text.	footer¹ -> footer[1]
u	Defines underlined text.	<u>underlined text</u> -> underlined text

Supported entity characters:

Character	Description	Code

"	quotation mark	&quot;
'	single quote, apostrophe	&apos;
&	ampersand sign	&amp;
<	less than sign	&lt;
>	greater than sign	&gt;
	non-breaking space	
©	copyright sign	&copy;
®	registered trade mark sign	&reg;
™	Registered Trademark sign	&trade;

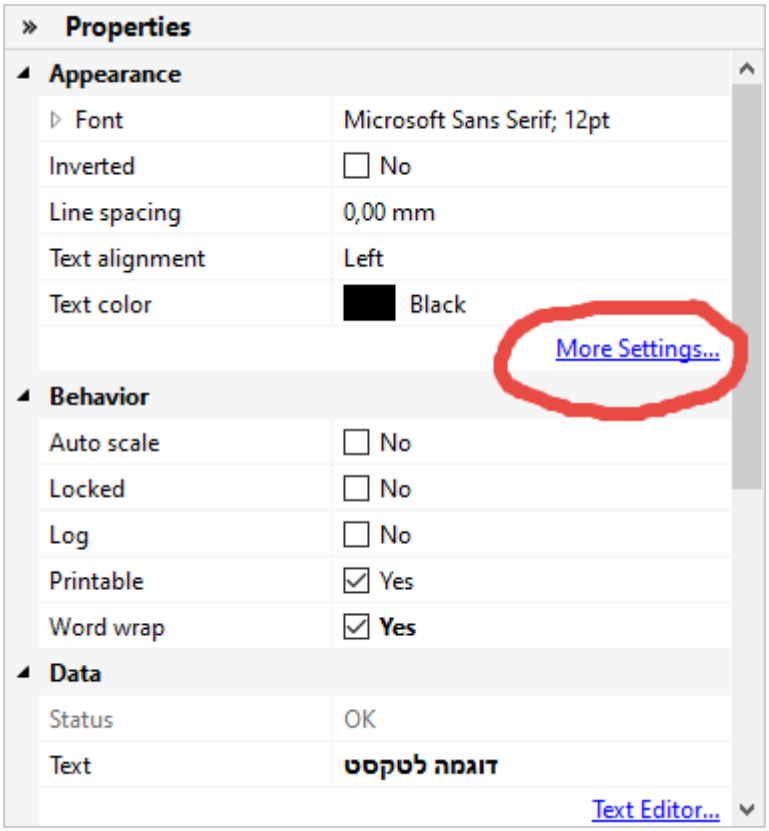
RTL Language Support

Labelstar Office supports languages written from right to left.

The default reading order in text fields is left to right. You can change the reading order to right-to-left, for example, if your data is in a right-to-left language such as Arabic or Hebrew.

To change the reading order, proceed as follows:

1. Select one or more text fields and click **More Settings**.
2. Activate **Right to left reading order**.



Supported Graphic and Vector Formats

- **ANIMATED GIF** - Graphics Interchange Format
- **BMP** - Standard Windows Bitmap
- **CUT** - Dr. Halo/Dr. Genius Clipboard Format
- **DDS** - Microsoft DirectDraw Surface Format
- **DIB** - Standard Windows Bitmap Format
- **DICOM** - Digital Imaging and Communications in Medicine
- **EMF** - Enhanced Windows Metaformat
- **EXIF** - Exchangable Image Format
- **EXR** - OpenEXR Format
- **FAX, G3** - Group 3 Raw Fax Format
- **GIF, Interlaced GIF** - Graphics Interchange Format
- **HDR** - High Dynamic Range Format
- **IFF** - Interchange Format
- **ICO (single and multi page)** - Icone Format
- **J2K, J2C** - JPEG-2000 Codestream
- **JB2, JBIG2** - Joint Bi-level Image Experts Group
- **JIF, JFIF** - JPEG File Interchange Format
- **JNG** - JPEG Network Graphics
- **JP2** - JPEG-2000 Format
- **JPEG, JPG, JPE** - Joint Pointgraphic Expert Group
- **JPEG progressive**
- **KOA** - KOALA Format
- **LBM** - Interchange File Format-Interleaved Bitmap
- **MNG** - Multiple-image Network Graphics
- **PBM** - Portable Bitmap File
- **PBM Raw** - Portable Bitmap BINARY
- **PCD** - Kodak Photo-CD file
- **PCT, PICT, PIC** - Macintosh PICT Format
- **PCX** - PC Paintbrush Format
- **PFM** - Portable Float Map
- **PGM** - Portable Graymap BINARY
- **PGM RAW** - Protable Graymap File
- **PSD** - Photoshop File
- **PNG** - Portable Network Graphics Format
- **PNM** - Portable Any Map
- **PPM** - Portable Pixmap File
- **PPM RAW** - Portable Pixmap BINARY
- **RAS** - Sun Raster Format
- **RAW camera image**
- **RAW memory bits** - RAW bitmap
- **RLE** - Standard Windows Bitmap format
- **SGI** - Silicon Graphics Image Format
- **TGA, TARGA** - TARGA Image Format
- **TIFF, TIF** - Tagged Image Format

- **TIFF Multi-page** - Multi-page Tagged Image Format
- **WBMP, WAP, WBM** - Wireless Bitmap
- **WEBP** - WebP Image Format
- **WMF** - Standard Windows Metaformat
- **XBM** - X Bitmap Format
- **XPM** - X Pixmap Format

Food Allergen Labelling

The new EU Regulation 1169/2011 on the provision of food information to consumers changes existing legislation on food labelling including:

- Mandatory nutrition information on processed foods.
- Mandatory origin labelling of unprocessed meat from pigs, sheep, goats and poultry.
- Highlighting allergens e.g. peanuts or milk in the list of ingredients.
- Better legibility i.e. minimum size of text.
- Requirements on information on allergens also cover non pre-packed foods including those sold in restaurants and cafés.

The new rules will apply from 13 December 2014. The obligation to provide nutrition information will apply from 13 December 2016.

Foods that need to be labelled on pre-packed foods when used as ingredients are:

- **Cereals containing gluten** such as wheat, rye, barley, oats, spelt or khorasan
- **Crustaceans** for example prawns, crabs, lobster, crayfish
- **Eggs**
- **Fish**
- **Peanuts**
- **Soybeans**
- **Milk**
- **Nuts** such as almonds, hazelnuts, walnuts, cashews, pecan nuts, Brazil nuts, pistachio nuts, macadamia (or Queensland) nuts
- **Celery** (including celeriac)
- **Mustard**
- **Sesame seeds**
- **Sulphur dioxide and sulphites** (>10mg/kg or 10mg/l)
- **Lupin**
- **Mollusc** for example clams, mussels, whelks, oysters, snails and squid

Sample

This sample shows how you can create a database label, using the variable $ReplacePattern, to automatically highlight certain words.

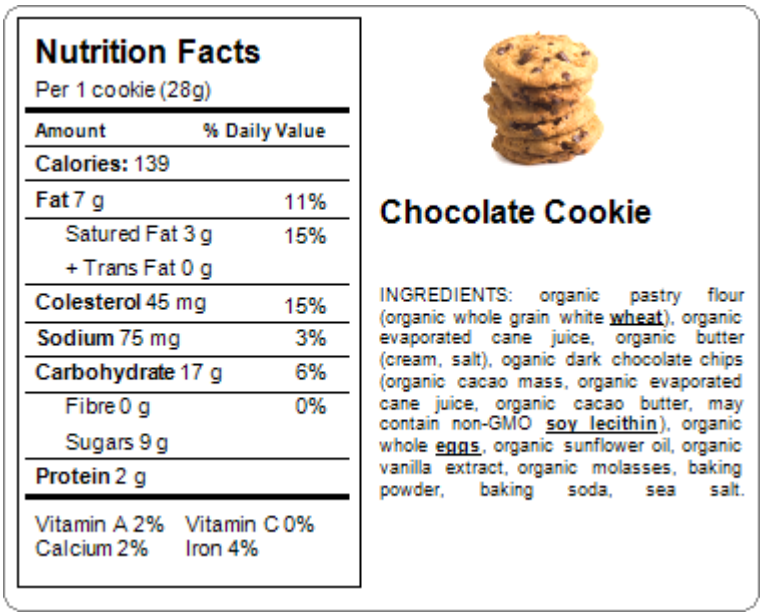
Example

The sample data can be found in the installation folder in the *Samples\Allergens* directory or can be downloaded here.

Allergens.txt This file contains the list of allergens that are being emphasized.
Cookies.accdb Microsoft Access database with the ingredients lists.
Cookies.lbex Label definition
Import Cookies_accdb data connection.lbdx Import file for data connection *Cookies*.

To print the sample label, proceed as follows:

1. Open **Labelstar Office**.
2. Import data connection **Cookies**
   - Activate **Data Connections** view.
   - Right-click to open context menu and click **Import Data Connection**.
   - Navigate to samples folder and open **Import Cookies_accdb data connection.lbdx**.
3. Open and print sample label



Printing in an SAP Environment

Prerequisites

Label Printing Software: **Labelstar Office** Version 4.30 Build 1015 or later
Windows Printer Drivers: Carl Valentin Printer Drivers Version 2.3.1 or later
Start/Stop Characters: 0x5E/0x5F

Designing the Label on the PC

Open **Labelstar Office** Version 4.30 Build 1015 or later to design the label layout. To define SAP form fields use $SAPField variables to define named data fields where SAPscript ITF will insert variable data. Use the SAP R/3 field names as field names (for example, VBAK_KUNNR).

For text fields, you must use printer-internal fonts, not TrueType fonts from Windows. This is because TrueType fonts are created in binary form, which SAPscript does not support.

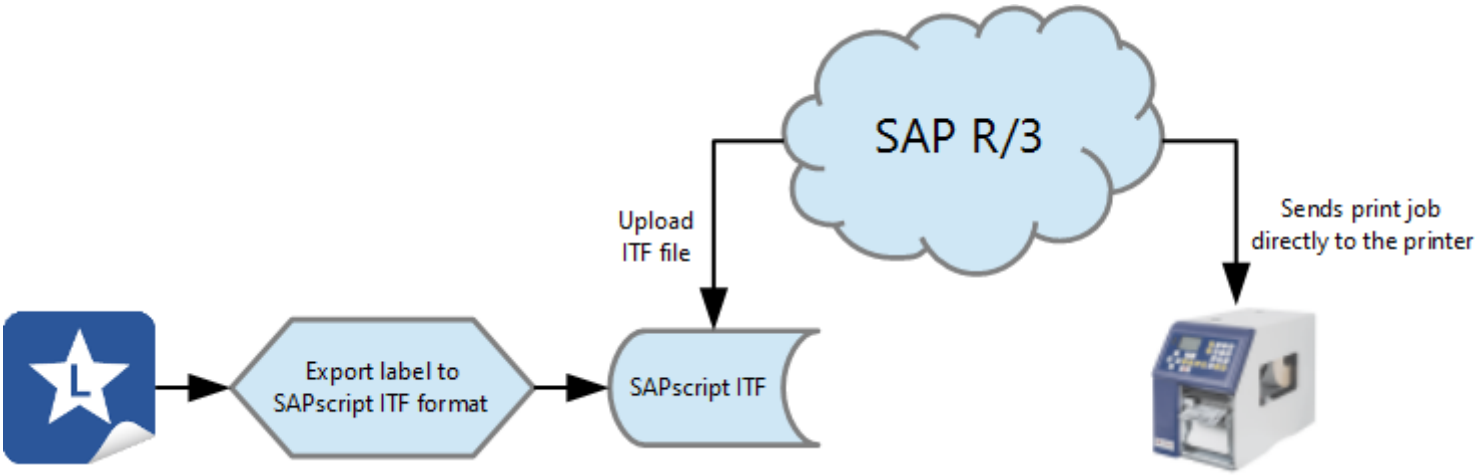
If there are images or text fields which use TrueType fonts on the label, two separate printer code template files are exported. One for the variable data and the other containing formatting information and binary data. Only the file for variable data will be uploaded in SAPscript ITF. The other file is stored on the memory card.

Exporting the Label to SAPscript ITF Printer Code Template File

1. Click the **File** tab, and then click **Export**.
2. Click **Create SAPScipt ITF File**, and the on the right, click **Create ITF**.
3. In the **Browse** dialog box, select a name and location for the file.

Upload the ITF File to SAPscript

Upload the ITF file, generated with **Labelstar Office**, using the SAP R/3 Windows Client. For this, the upload function available for the standard text editor (Transaktion SO10) is used. In this case, however, the editor is only used as a "clipboard" for the file until you insert the file in a Form Window.



Printer-Internal Print Jobs

Note

This feature requires Carl Valentin Printers.

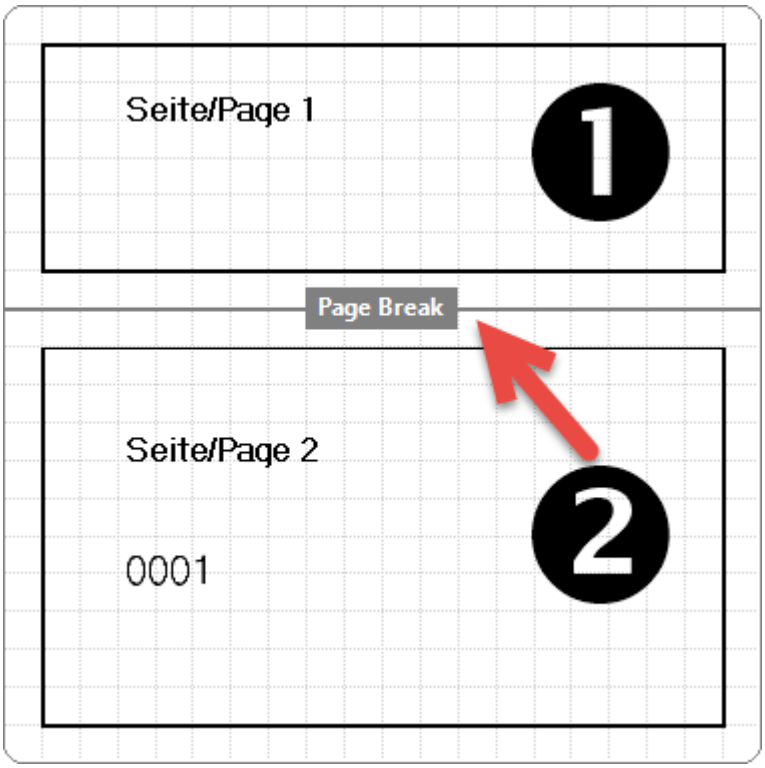
What is a Print Job?

With job printing, it is possible to define different pages within a print cycle and to print them in any order.

Note

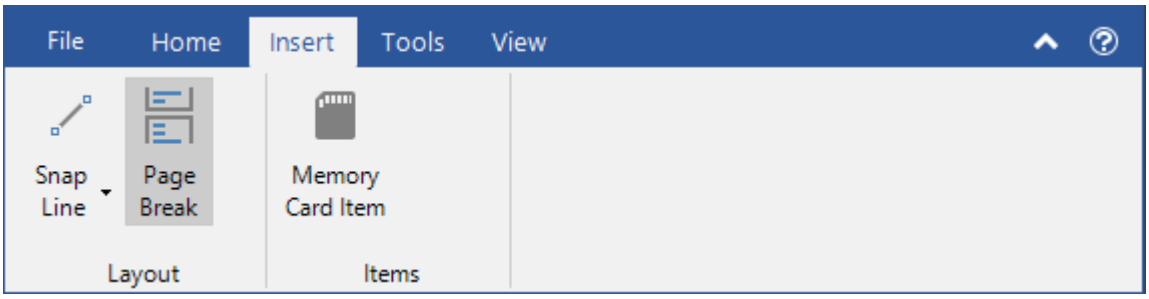
When defining the pages, please note that the available memory is limited. Currently, depending on the page size, a maximum of nine pages can be defined.

To separate a label into pages, one or more page breaks must be inserted.



To insert a page break, proceed as follows:

1. Click **Insert** > **Page Break**



   The new page break is inserted at the top of the label.
2. Drag the page break to the desired place on the label.

Sample

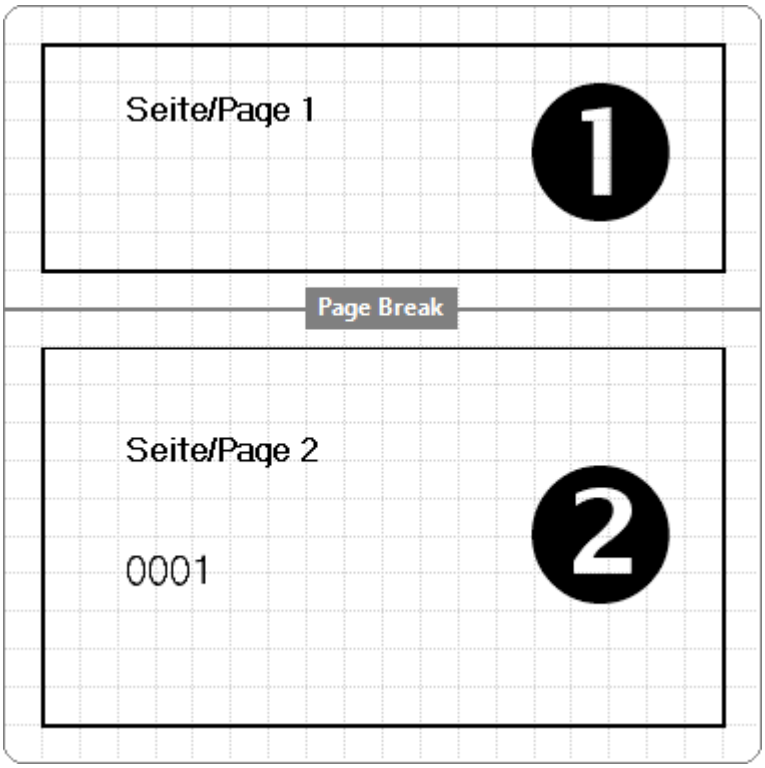
Note

This feature requires Carl Valentin Printers.

Example

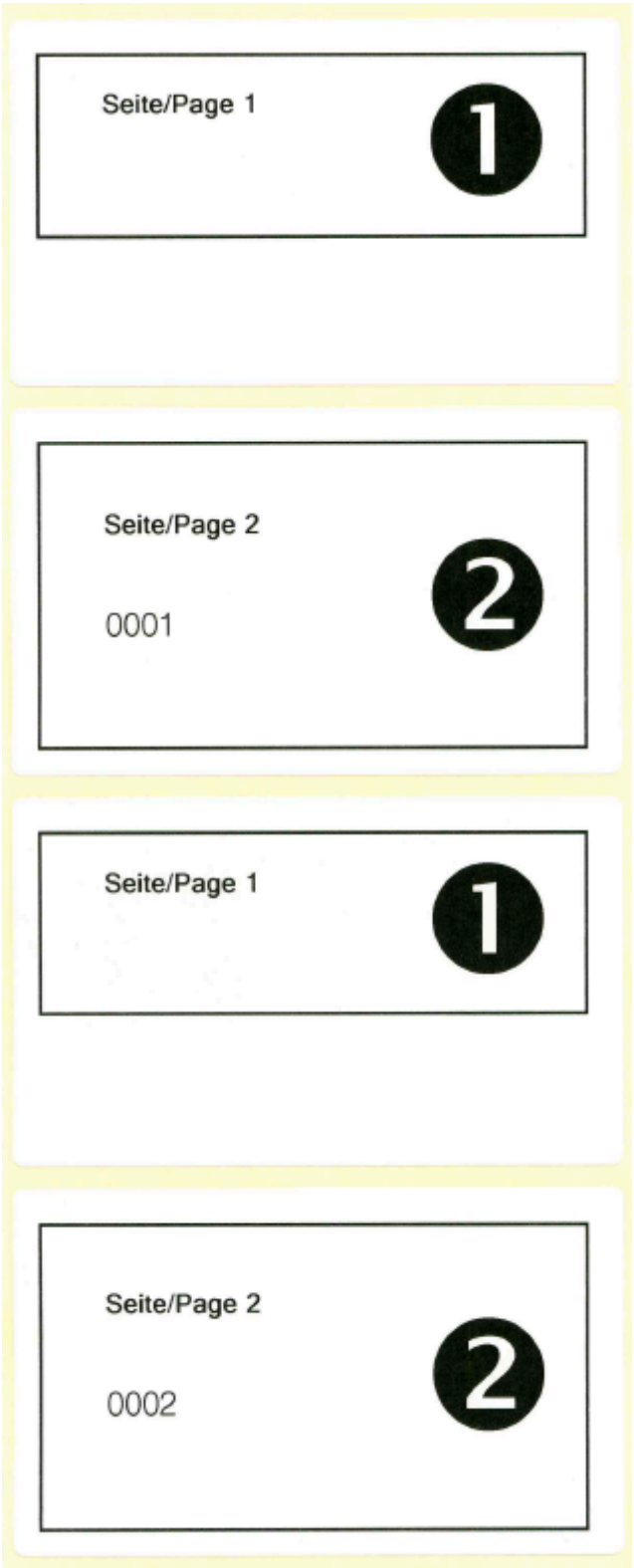
The sample label *PrintJob.lbex* can be found in the installation folder in the *Samples* directory or can be downloaded here.

Label Definition

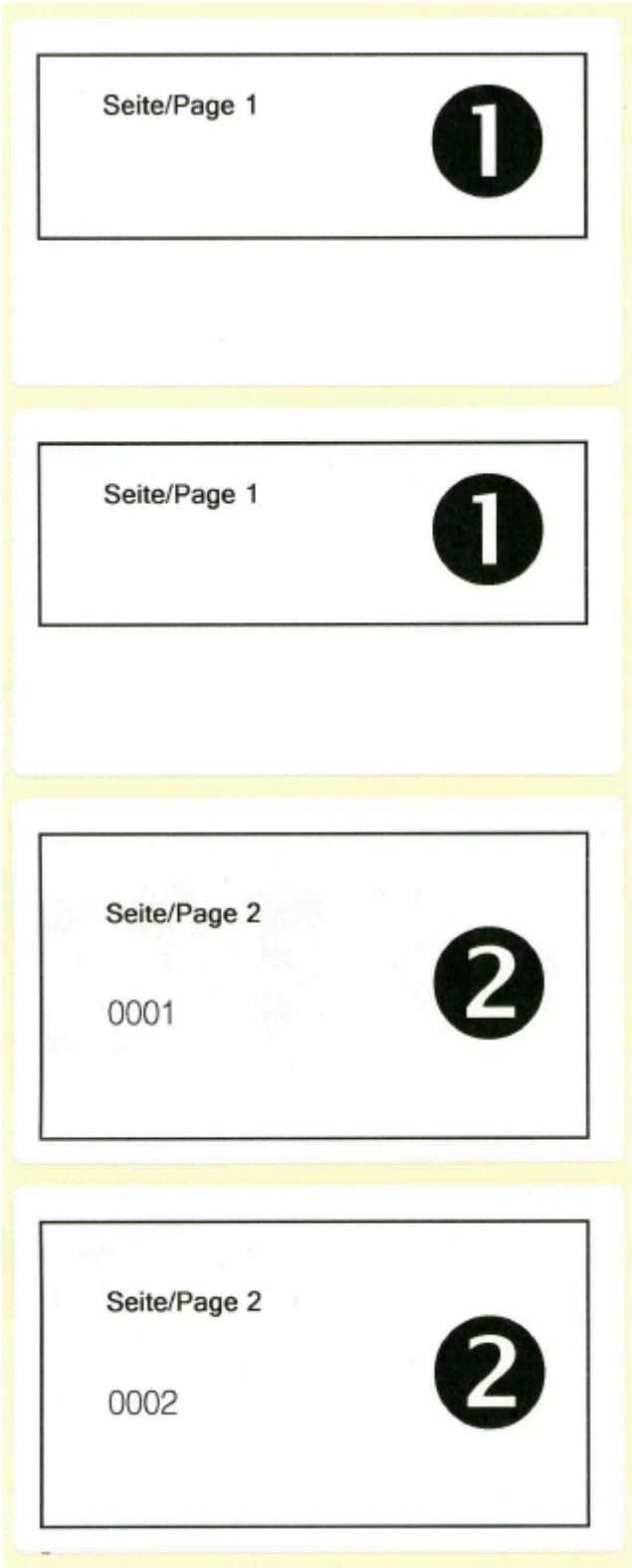


Printout

Collated



Not collated



Country Codes

Note

This list should only give an overview and has no claims to correctness and completeness.

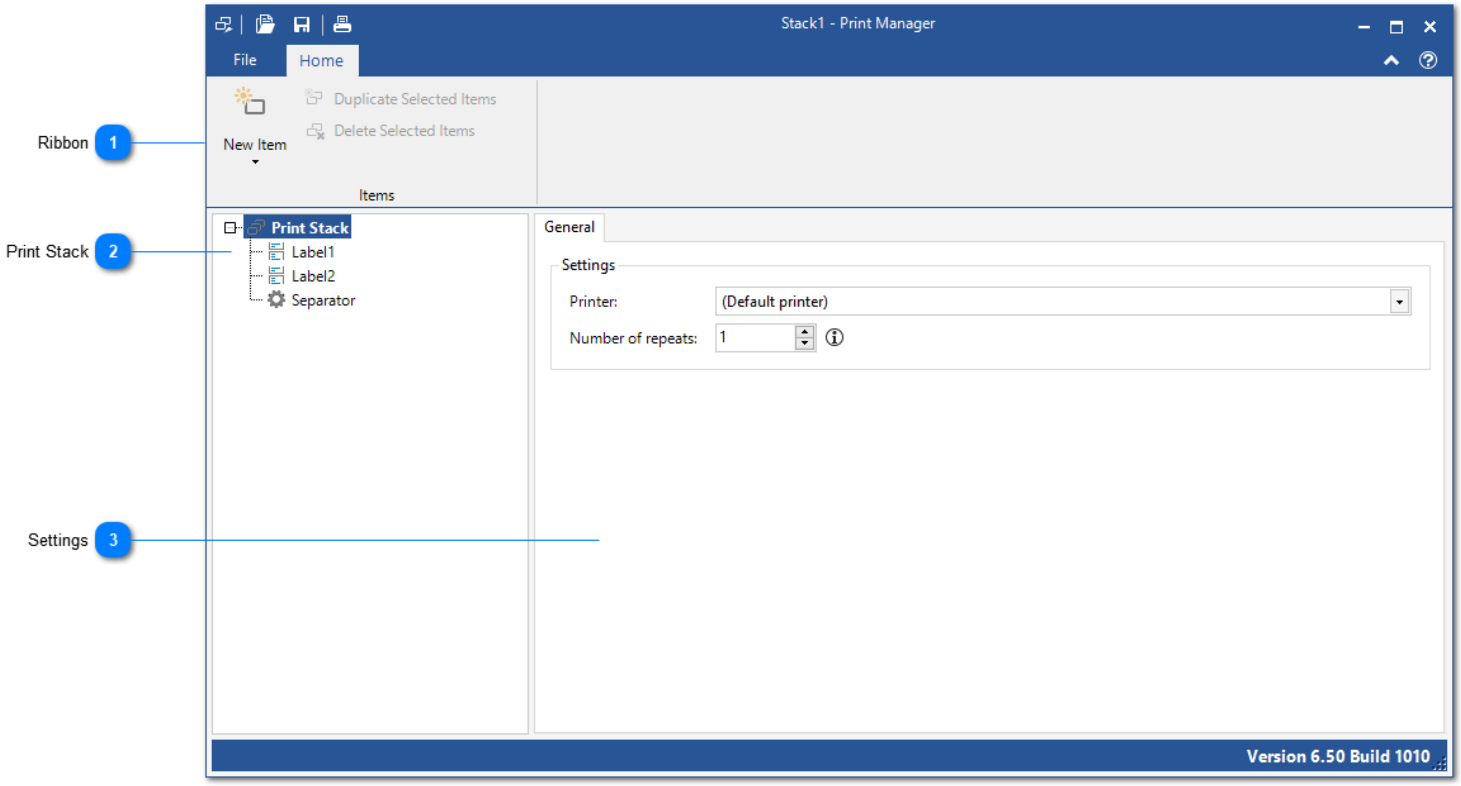
Country Code	Language	
zh-CN	Chinese (People's Republic of China)	中文(中华人民共和国)
zh-Hans	Chinese (simplified)	中文(简体)
da	Danish	dansk
de	German	Deutsch
de-DE	German (Germany)	Deutsch (Deutschland)
de-LI	German (Liechtenstein)	Deutsch (Liechtenstein)
de-LU	German (Luxembourg)	Deutsch (Luxemburg)
de-CH	German (Switzerland)	Deutsch (Schweiz)
en	English	English
en-GB	English (Great Britain)	English (United Kingdom)
en-US	English (US)	English (United States)
fi	Finnish	suomi
fr	FrenchFrench	français
fr-BE	French (Belgium)	français (Belgique)
fr-FR	French (France)	français (France)
fr-LU	French (Luxembourg)	français (Luxembourg)
fr-CH	French (Switzerland)	français (Suisse)
el	Greek	ελληνικά
he	Hebrew	עברית
nl	Dutch	Nederlands
nl-BE	Dutch (Belgium)	Nederlands (België)
nl-NL	Dutch (Netherlands)	Nederlands (Nederland)
it	Italian	italiano
it-IT	Italian (Italy)	italiano (Italia)
it-CH	Italien (Switzerland)	italiano (Svizzera)
ja	Jaamese	日本語
pl	Polish	polski
pt-BR	Portuguese (Brazil)	Português (Brasil)
pt-PT	Portuguese (Portugal)	Português (Portugal)
ru	Russian	русский
es	Spanish	español
es-ES	Spanish (Spain)	español (España)
cs	Czech	čeština
tr	Turkish	Türkçe
hu	Hungarian	magyar

For a detailed list of all country codes, click here.

Print Manager

The **Print Manager** can define print stacks with several labels and print them in one step. This allows recurring print processes to be combined in one file and printed with one command.

The print stacks can be printed directly from the **Print Manager** or from the Quick Print application.



1 Ribbon

2 Print Stack

This displays all elements to be printed. The position of the individual elements within the list can be changed by clicking and moving (drag & drop).

3 Settings

This area displays the settings which can be changed for the selected element.

See also

- Create print stack
- Print print stack
- Share programs

Create Print Stack

To insert an item into the print stack, proceed as follows:

1. Start the **Print Manager**.
2. In the **Items** group on the **Home** tab, click **New Item** and select one of the following entries:
   - **Label** Inserts a label file into the print stack.
   - **Label (Database)** Inserts a label into the print stack whose file name is generated from a database field.
   - **Separator Label** Prints a grey bar as separation label.
   - **Blade Cut** Performs a blade cut.
   - **Print Command** Transmits one or more print commands to the printer.

See also

- Print Manager
- Print print stack
- Share programs

Print Print Stack

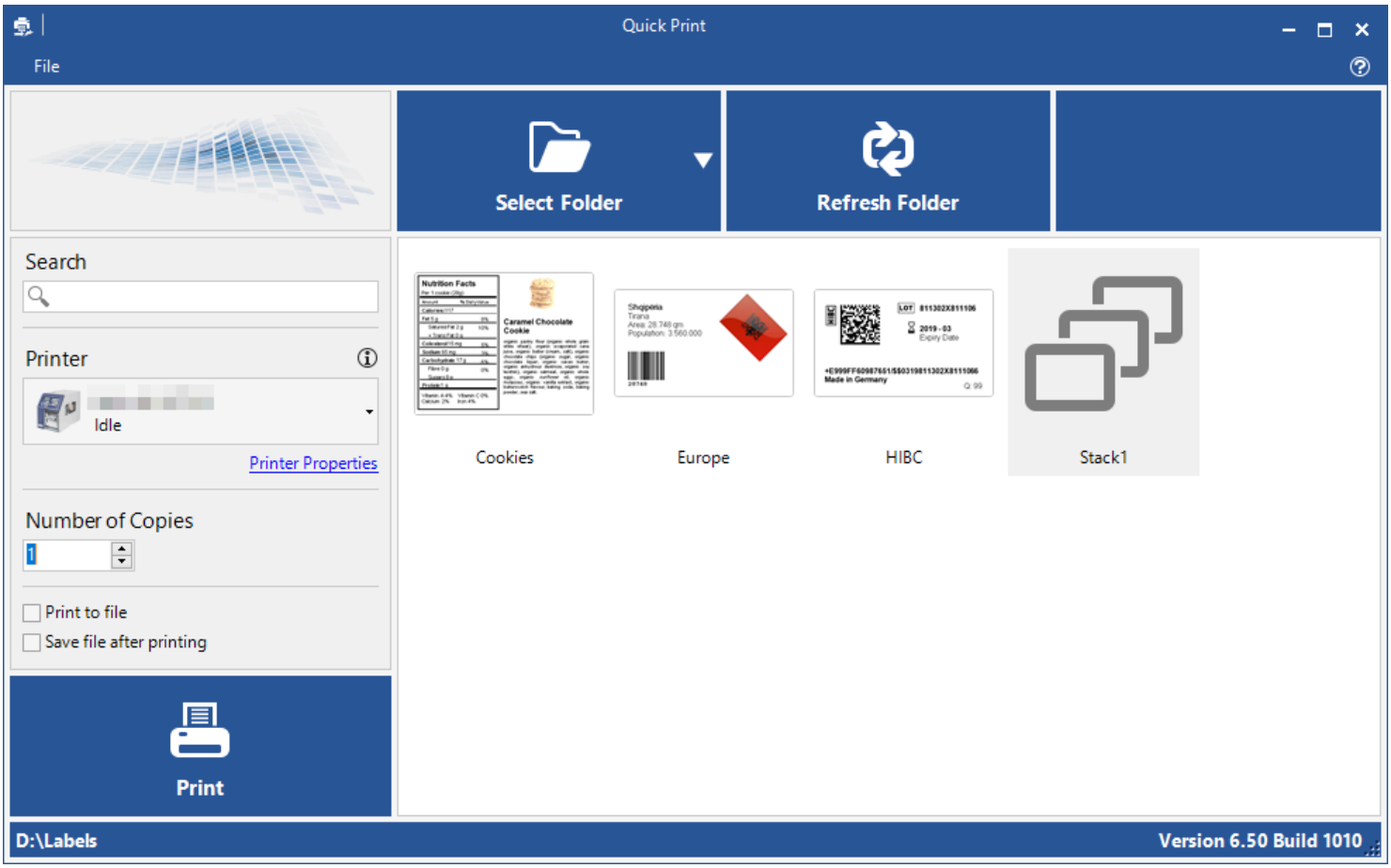
There are several ways to print a print stack.

Printing a print stack in the Print Manager

1. Start the **Print Manager**.
2. Create a new print stack or open an existing one.
3. Do one of the following actions:
   - Click **File**, and then click **Print**.
   - On the Quick Access Toolbar, click to start printing.

Printing a print stack in Quick Print

1. Start the Quick Print application.
2. Select the directory where the print job is stored.
3. Select the print stack and then click the **Print** button.

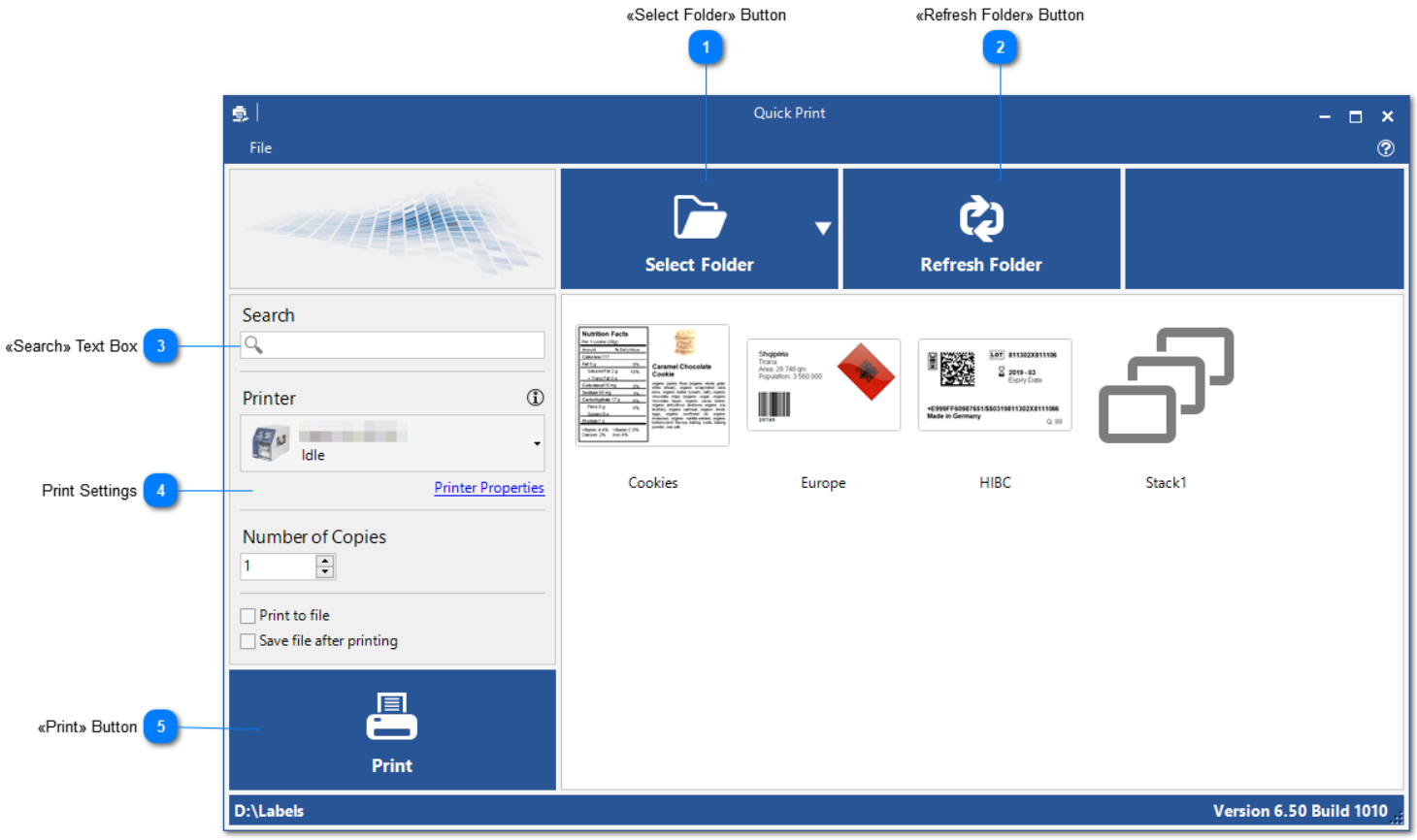


See also

- Print Manager
- Create print stack
- Share programs

Quick Print

Labelstar Office Quick Print allows you to view all the label (.lbex) files in a specified folder and even print them.



1 «Select Folder» Button

Select the directory in which the files are saved or click ▼ in order to be shown a list of recently used directories.

2 «Refresh Folder» Button

Refreshing previews (if a file was added to the folder for example).

3 «Search» Text Box

If you want to narrow the number of files shown, you can enter a search/filter term in the text box and press RETURN. Only files that contain the entered term will be displayed.

4 Print Settings

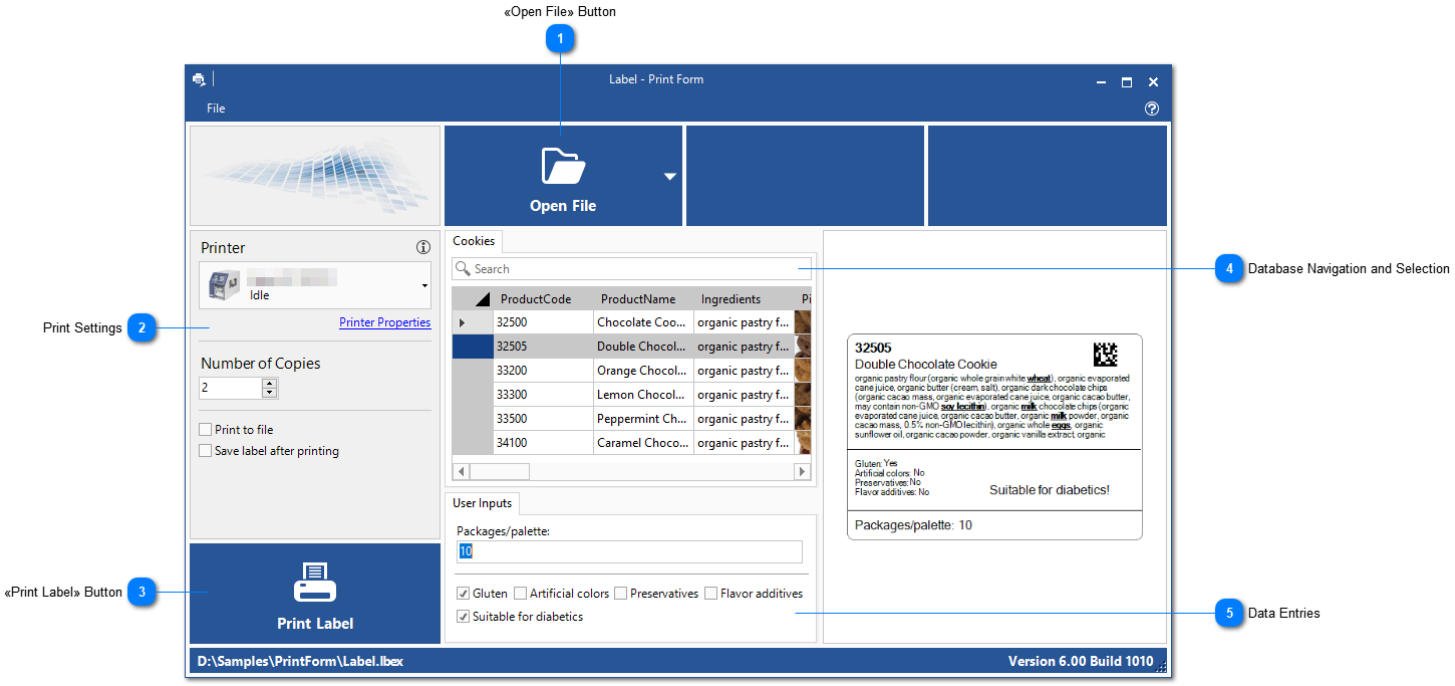
- **Printer** Select the printer on which the selected label should be printed. In order to change various print settings, click on **Printer Properties**.
- **Number of Copies** Enter the number of copies which should be printed. If a number of prints is saved for a label then the display will be adjusted and deactivated when the label is selected.
- **Print to file** If this option is activated, the print output is redirected into a file.
- **Save file after printing** If this option is activated, the label will be saved after printing. This is important, for example, if the end value of a counter should be retained.

5 «Print» Button

Prints the selected label or print job.

Print Form

Labelstar Office Print Form is a "all-in-one" form. The printing form includes all required user interface objects: data entry fields, database navigation and selection, print settings and a print preview of the label. That optimizes the printing process and reduces the risk of errors.



1 «Open File» Button

Browse for the file you want to open or click ▼ to load a file you have been working with recently.

2 Print Settings

- **Printer** Select the printer on which the selected label should be printed. In order to change various print settings, click on **Printer Properties**.
- **Number of Copies** Enter the number of copies which should be printed. If a number of prints is saved for a label then the display will be adjusted and deactivated when the label is selected.
- **Print to file** If this option is activated, the print output is redirected into a file.
- **Save label after printing** If this option is activated, the label will be saved after printing. This is important, for example, if the end value of a counter should be retained.

3 «Print Label» Button

Prints the label on the selected printer.

4 Database Navigation and Selection

When you select a record the label preview is updated to reflect the changes.

5 Data Entries

After each change of a user input, the label preview is updated.

Folder Monitor

The automated printing of **Labelstar Office** is designed for businesses that want to increase efficiency by automating their printing process. Printing is done in the background and can be started from your own application. This makes it possible to control and automate even very complex printing processes completely from your own application.

You can use the Folder Monitor application to easily create and manage your system integrations.

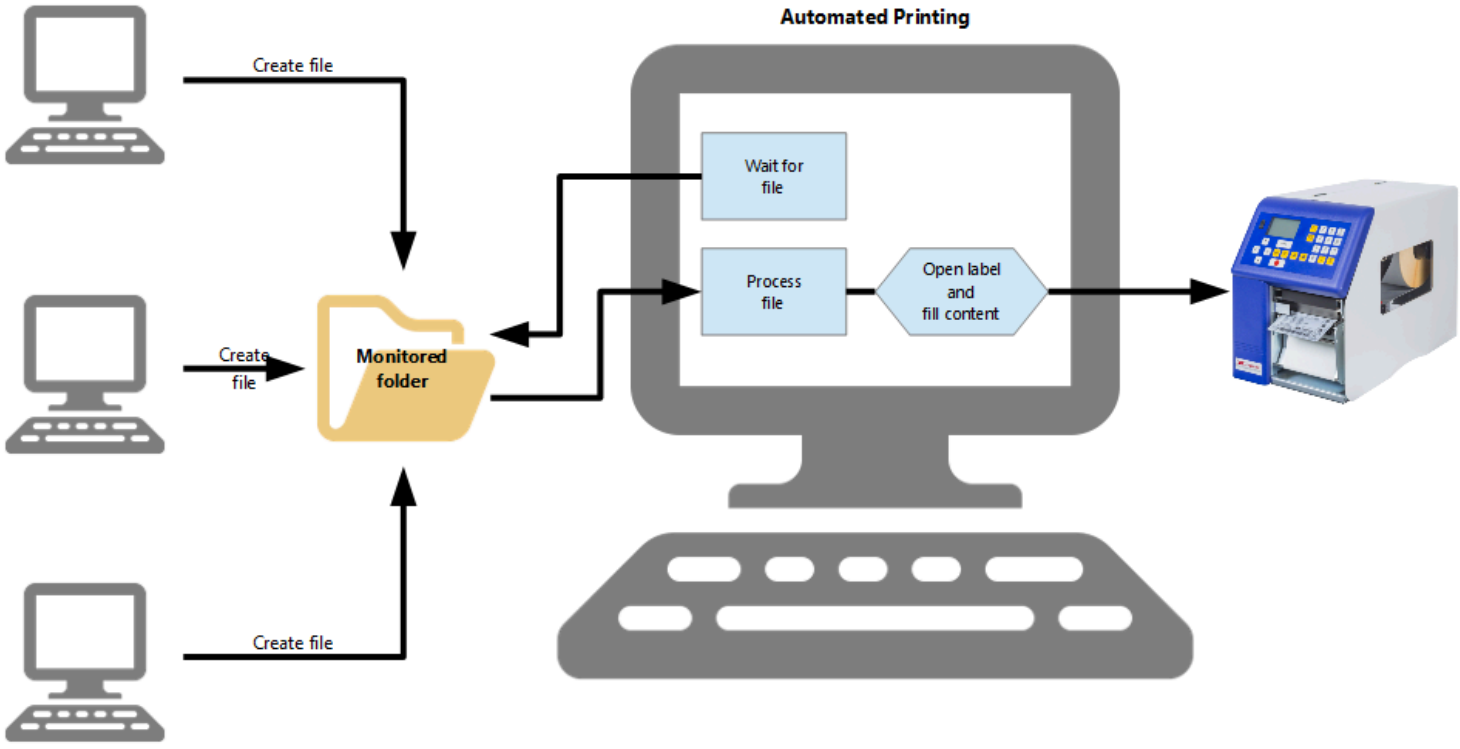
Note

Starting with program version 6.50, the automated printing is executed as a Windows service.

Example

The following application example illustrates the fundamental concept of data exchange between your application and the automated printing of **Labelstar Office**.

A third-party system (e.g. SAP) creates a file in LSO XML file format or in CSV file format with the required data and copies the file to one of the monitored folders. If a change is detected in any of the monitored folders, the application evaluates the file, fills the associated label layout and transmits the data to the printer.



See also

- Configure automated printing
- Supported file formats
- Share programs

Concepts & Terms

The automated printing of **Labelstar Office** can be installed on a Windows server or workstation and monitors the file activity on that computer.

The automated printing of **Labelstar Office** consists of two parts: a graphical user interface which can be used to configure and manage the folder monitoring, and a background process (**Folder Monitor Service**) that processes the notification files and controls the printing process. The **Folder Monitor Service** is invisible and does not have its own user interface.

Folder Monitor Service

The **Folder Monitor Service** is the part of **Labelstar Office** that perfoms the monitoring of the file activity on the local computer. The service is set up to run automatically when Windows starts. The **Folder Monitor** application does not need to be running in order for monitoring to take place.

See also

- Configure automated printing
- Supported file formats
- Share programs

Configuration

The **Folder Monitor** can be used to configure and manage the automated printing of **Labelstar Office**.

To launch the program, go to the Windows main menu and select the **Programs > Labelstar Office > Folder Monitor** item.

Select a new folder to monitor

1. Select **Folders** tab.
2. To select a new folder, use one of the following procedures:
   - **Quick Add:** Click  , browse for the directory you want to monitor and then click **Add**.
   - On the **Folders** tab, in the **New** group, click **New Folder**.

Start/Stop Folder Monitor Service

In order for the changes made in the **Folder Monitor** application to be taken into account in the **Folder Monitor Service**, the service must be restarted. On the Quick Access Toolbar, click ■ to stop monitoring and ▶ to start monitoring.

Once the **Folder Monitor Service** has been started, it will be set up to run automatically when Windows starts. The **Folder Monitor** application does not need to be running in order for monitoring to take place.

Note

In order to start or stop the service from the **Folder Monitor** application, the program must be executed as administrator.

See also

- Supported file formats
- Share programs

Supported File Formats

Two different file formats are currently supported:

- LSO XML File Format
- CSV File Format

LSO XML File Format

Sample XML File (UTF-8 encoded)

```
<?xml version="1.0" encoding="utf-8"?>
<LSO version="1.0">
   <layout>D:\Label.lbex</layout>
   <printer>Vario III 107/12</printer>
   <copies>2</copies>
   <field name="Text1">Example</field>
   <field name="Text2" />
   <field name="EAN">1234567890123</field>
</LSO>
```

Note

All LSO XML files use **UTF-8** (`encoding="uft-8"`) character encoding by default.
If you want to create or change the content of an LSO XML file you should use a text editor like Notepad that supports UTF-8 encoding and make sure when saving the file that **UFT-8** encoding is selected. When saving the edited file by using a non-UTF-8 compatible editor or with a different character encoding, the encoding breaks, resulting in a non-valid XML file.

More examples are found here.

Supported XML Tags

Tag	Description	Attributs	Parent-tag	Sub-tags
LSO	XML top level element.	*version* - 1.0 (Version number of the XML file format)	None	label layout printer copies field select
label	This tag defines a label. Using this tag multiple labels can be printed consecutively (see also *Sample2.xml*).	None	LSO	layout printer copies field select
layout	This tag contains the file name of the label. If no layout is defined, the layout defined in the **Monitored Folder** preferences will be used.	None	LSO label	None
printer	This tag contains the printer name. If no printer name is defined, the printer defined in the **Monitored Folder** preferences or the printer stored with the label will be used.	None	LSO label	None
copies	This tag contains the number of copies to print. If no number of copies is defined, the number of copies defined in the **Monitored Folder** preferences or the number of copies stored with label will be used.	None	LSO label	None
field	This tag contains the field content of a specific field.	*name* - Field name	LSO label	None
select	Selects a record. The value to be specified is defined by the **Record Selection** in the data connection.	*name* - Data connection name	LSO label	None

More information about XML can be found on the Internet (for example, enter the keyword "XML tutorial" in your search engine).

CSV File Format

Sample CSV File (First Row Contains Field Names)

```
Layout;Printer;Copies;Text1;Text2;EAN
"D:\Label.lbex";"Vario III 107/12";2;Example;;1234567890123
```

Fixed Column Names

Column name	Description
Layout	This column contains the file name of the label. If no layout is defined, the layout defined in the **Monitored Folder** preferences will be used.
Printer	This column contains the printer name. If no printer name is defined, the printer defined in the **Monitored Folder** preferences or the printer stored with the label will be used.
Copies	This column contains the number of copies to print. If no number of copies is defined, the number of copies defined in the **Monitored Folder** preferences or the number of copies stored with label will be used.

Sample

This sample shows how labels can be printed automatically using **Labelstar Office**.

Example

The sample files can be found in the installation folder in the *Samples\Folder Monitor* directory or can be downloaded here.

Images Image folder
Label.lbex Label
Sample1.xml This file prints the label *Label.lbex* twice with the same content.
Sample2.xml This file prints the label *Label.lbex* three times with different content.

To run the sample, proceed as follows:

1. Create a new folder, *C:\Data*
2. Go to the Windows main menu and select the **Programs > Labelstar Office > Folder Monitor (execute as administrator)** item.
   The program **Folder Monitor** opens.
3. **Quick Add:** Click ..., browse for *C:\Data* and click **Add**.
4. On the Quick Access Toolbar, click ▶ to start monitoring.
5. Go to the sample folder and copy *Sample1.xml* and *Sample2.xml* to *C:\Data*.
6. The labels are printed on the current default printer.

Sample1.xml

This file prints *Label.lbex* twice with the same content.

File Content

```
<?xml version="1.0" encoding="utf-8"?>
<LSO version="1.0">
  <layout>%InstallDir%\Samples\Folder Monitor\Label.lbex</layout>
  <copies>2</copies>
  <field name="OrderNumber">P459730</field>
  <field name="BatchNumber">003</field>
  <field name="Deutsch">Zange</field>
  <field name="English">Pliers</field>
  <field name="Francais">Pince</field>
  <field name="Image">%InstallDir%\Samples\Folder Monitor\Images\Pliers.png</field>
  <field name="Barcode">P459730003</field>
</LSO>
```

Printout

See also

- Sampel
- Folder Monitor

Sample2.xml

This file prints *Label.lbex* three times with different content.

File Content

```
<?xml version="1.0" encoding="utf-8"?>
<LSO version="1.0">
   <label>
       <layout>%InstallDir%\Samples\Folder Monitor\Label.lbex</layout>
       <copies>1</copies>
       <field name="OrderNumber">H876324</field>
       <field name="BatchNumber">002</field>
       <field name="Deutsch">Hammer</field>
       <field name="English">Hammer</field>
       <field name="Francais">Marteau</field>
       <field name="Image">%InstallDir%\Samples\Folder Monitor\Images\Hammer.png</field>
       <field name="Barcode">H876324002</field>
   </label>
   <label>
       <layout>%InstallDir%\Samples\Folder Monitor\Label.lbex</layout>
       <copies>1</copies>
       <field name="OrderNumber">C128703</field>
       <field name="BatchNumber"/>
       <field name="Deutsch">Messschieber</field>
       <field name="English">Caliper</field>
       <field name="Francais">Pied à coulisse</field>
       <field name="Image">%InstallDir%\Samples\Folder Monitor\Images\Caliper.png</field>
       <field name="Barcode">C128703</field>
   </label>
   <label>
       <layout>%InstallDir%\Samples\Folder Monitor\Label.lbex</layout>
       <copies>1</copies>
       <field name="OrderNumber">B080213</field>
       <field name="BatchNumber">001</field>
       <field name="Deutsch">Schraube</field>
       <field name="English">Bolt</field>
       <field name="Francais">Vies</field>
       <field name="Image">%InstallDir%\Samples\Folder Monitor\Images\Bolt.png</field>
       <field name="Barcode">B080213001</field>
   </label>
</LSO>
```


Printout

See also

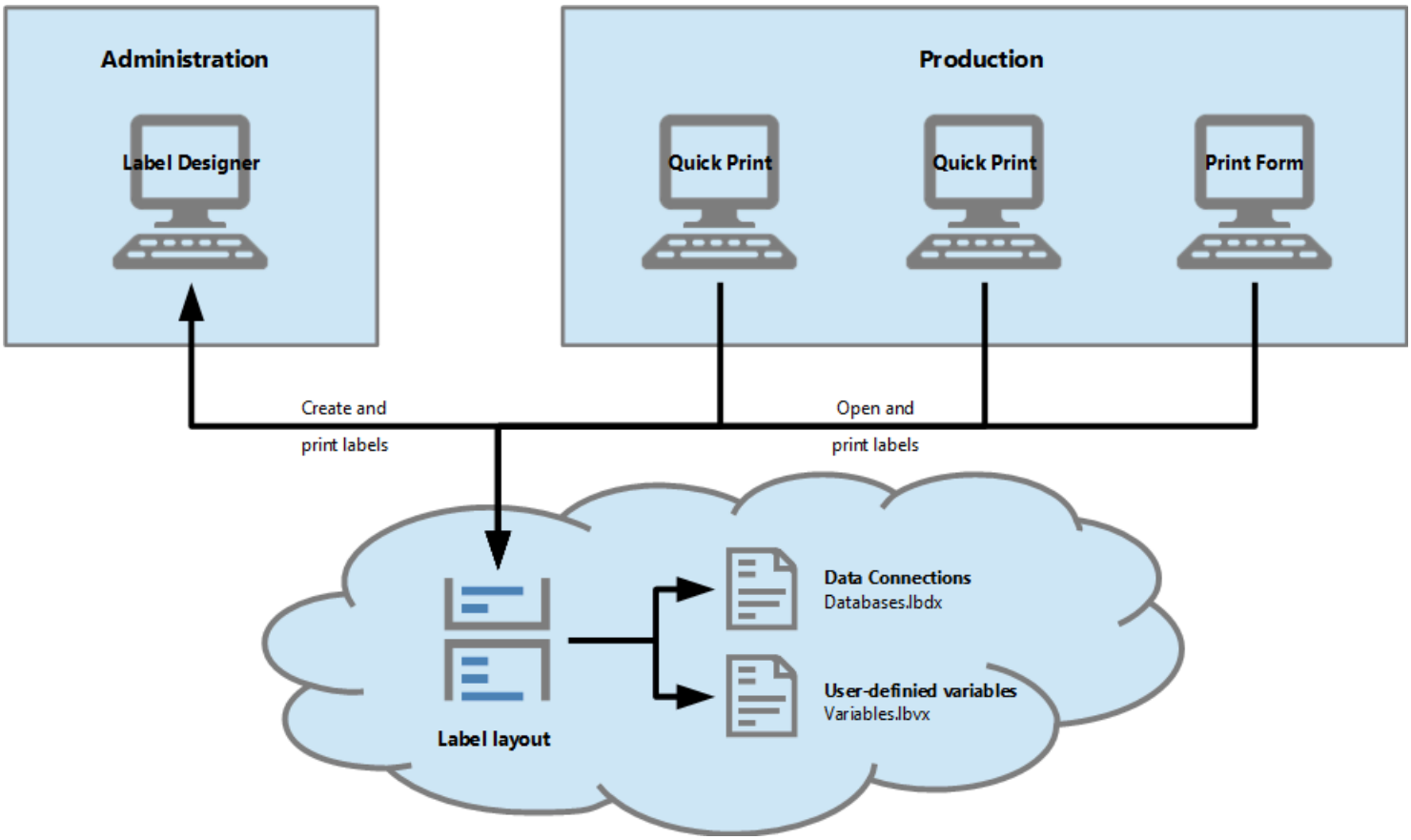
- Sample
- Folder Monitor

Share Programs

Application Situation

In a company, only certain workstations should be able to create and changes labels. From other workstations, e.g. in the production, it should only be possible to open and print the labels, without the possibility to change the label layouts.

For this application situation, **Labelstar Office** provides different program components. With the Label Designer labels can be created, changed and printed. At the workstations where the labels are only to be printed, the programs Quick Print and Print Form can be used.



Requirements

To ensure that the labels are printed correctly from all workstations, a few prerequisites must be met.

If database fields are defined or user-defined variables are used on the labels, it must be ensured that all workstations can access the required files. In particular to the definition files used by **Labelstar Office** in which the data connections (*Databases.lbdx*) and the user-defined variables (*Variables.lbvx*) are stored. By default, these files are stored in the directory *%ProgramData%\Labelstar Office* on the local machine.

To share variables and data connections, proceed as follows:

Note

You only need to change these settings if you use database fields and/or user-defined variables on your labels.

Label Designer

1. Open **Label Designer**.
2. Activate **Data Connections** view.
3. Select and then click **Save As**.
4. Browse for the new location. Make sure that all workstations have access (read and write) to this location.

If you use user-defined variables on your labels, activate **Variables** view and repeat steps 3 and 4.

Quick Print/Print Form

1. Open the application.
2. Click **File > Options** and select **File Locations** tab.
3. Change the file locations of the **Database File** and/or the **Variable File** to the same locations used in the Label Designer.
4. Restart program.

Tools

Labelstar Office provides various tools to manage program data.

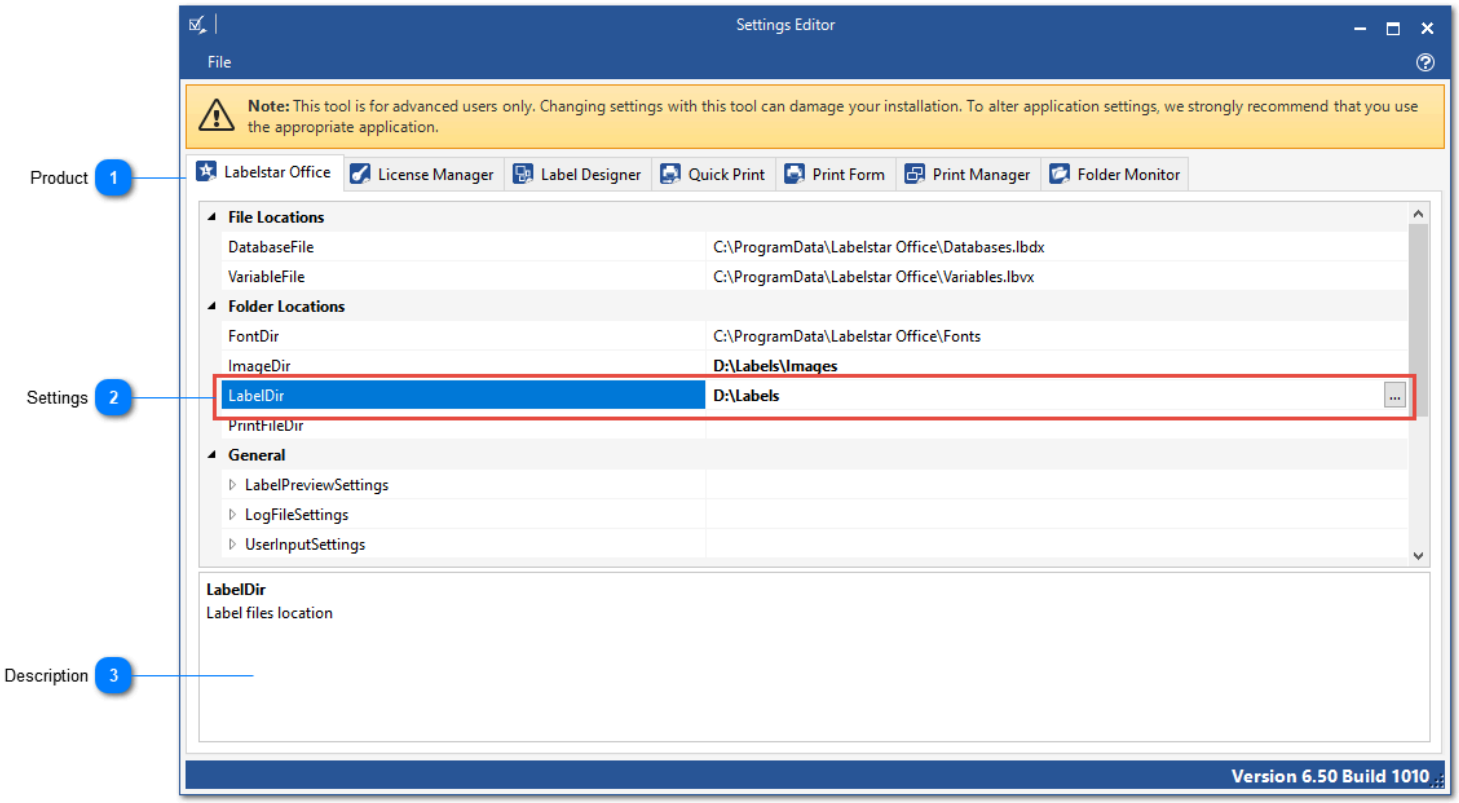
- **Settings Editor**
  With this tool, the internal **Labelstar Office** settings files can be changed.

Settings Editor

The **Settings Editor** is a tool for viewing, editing and managing the **Labelstar Office** settings files. The **Labelstar Office** settings files are text based XML files. They are found in the designated Application Data directory: *%ProgramData%\Labelstar Office*.

Although the **Settings Editor** enables you to inspect and modify **Labelstar Office** settings, normally you do not need to do so, and making incorrect changes can break your system. We strongly recommend that you use each **Labelstar Office's** product to modify its own settings, rather than attempt to edit the settings directly in the **Settings Editor**.

To launch the program, go to the Windows main menu and select the **Programs > Labelstar Office > Tools** item, and then double-click **Settings Editor**.



1 Product

Select the **Labelstar Office's** product whose settings you want to change.

2 Settings

Each product has a number af associated settings that you can modify. Each setting consists of a name (not editable) and a value.

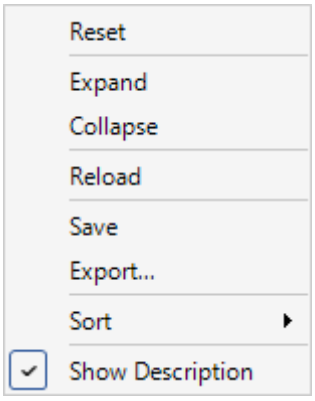
Right-clicking displays a context menu with further options.

3 Description

Provides a description of the selected setting.

Context Menu

When you right-click on a selected setting, this context menu becomes available. It enables you to perform operations on the settings.



Reset Resets selected setting to default.

Expand Expands selected category.

Collapse Collapses selected category.

Reload Reloads settings file.

Save Saves settings file.

Export Saves settings file to a different location.

Sort The settings can be viewed **Categorized** or **Alphabetically**.

Show Description Toggles the display of the description pane.

See also

- Settings Editor

Installation

Downloading and installing Labelstar Office

1. Download the current version of **Labelstar Office** from here.
2. Start the installation by double-clicking on the file you have just downloaded.

Licensing

In this section we aim to provide you with all the information you need to understand how your license works, and what to do if you experience any problems.

Got problems with your license? Contact us now.

Do I have to renew the license annually?

No. Licenses for the software are perpetual.

Do I have to pay extra for the updates within the same version of the software?

No, you have the right to update the application with all point upgrades regardless of the date of their release at no additional cost. For example, if you are a paid user of **Labelstar Office 6.0**, you can update with all installers versioned 6.x on top of the current version at no cost. But at the same time, if you like the features of the newer 7.x version, you need to pay for the upgrade.

How many licenses do I need?

The application is licensed per computer. You need to purchase a license for every computer that will run **Labelstar Office**. For more information, see License Agreement.

How can I upgrade from version 6.x?

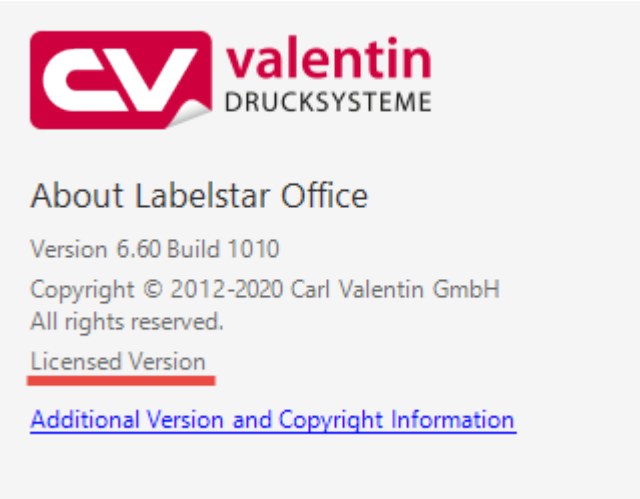
The upgrade of **Labelstar Office** software is provided at additional costs therefore you need to buy new license keys. For more information, see Upgrade Instructions.

How do I activate my software?

Once the order is completed, you will receive one or more license keys which will upgrade the program from the demo version to the full version. For more information see Activation.

How can I notice if my software was already activated?

Click **Help** on the **File** tab. Check **About** to see if you are working with a trial or licensed version.



How can I check, delete or change a license key?

Click **Help** on the **File** tab and then **Deactivate License**.

What is a trial version?

A trial version allows you to test the program. In the trial version some functions are limited, all e are replaced by x and all 0 by 5. All images are marked with a watermark. In the trial version certain functions or programs are possibly activated which are not included in the scope of delivery of the product bought by you. After you have entered a valid license key only the programs and features bought by you are shown.

Activation

Prior to activating **Labelstar Office**, you must install the trial version first. Furthermore, once the trial is installed you do not need to reinstall the program or use any other installer to activate the full version. Simply start the program and wait until the first dialog box appears. Now enter the license key and click **Save License Key**.

See also

- Licensing

Upgrade Instructions

IMPORTANT

If you are upgrading from **Labelstar Office 6.x** to **Labelstar Office 7.x**, you must obtain a new license key. Please contact your local dealer to purchase an upgrade license.

To upgrade from Labelstar Office 6.x to Labelstar Office 7.x, proceed as follows:

1. Download the current version of **Labelstar Office** from here. Ensure that you have valid license keys before proceeding with the upgrade. **Labelstar Office 7.x** uses different license keys than **Labelstar Office 6.x** and your old license keys will not be recognized by the program once you upgrade. If you do not have valid license keys **Labelstar Office** will run in trial mode.

   Please contact your local dealer for more information about upgrading your license.
2. Run the installer for the new version. The installer will detect the presence of an old version of **Labelstar Office** and prompt you to allow the upgrade to continue.
3. Follow the installer prompts.
4. Start the program and wait until the first dialog box appears. Now enter the new license key and click **Save License Key**.

See also

- Licensing

License Agreement

IMPORTANT - READ CAREFULLY

This End User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and the manufacturer ("MANUFACTURER") for the software product. The software product includes computer software and may include associated media, printed materials, and "online" or electronic documentation ("SOFTWARE PRODUCT"). By installing, copying or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, you are not authorised to install and you may not use the SOFTWARE PRODUCT.

SOFTWARE PRODUCT LICENSE
The SOFTWARE PRODUCT is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties.

The SOFTWARE PRODUCT is licensed, not sold.

GRANT OF LICENSE
This EULA grants you the following rights:

- **Application software** You may install or use one copy of the SOFTWARE PRODUCT, or any prior version for the same operating system, on a single computer. The primary user of the computer on which the SOFTWARE PRODUCT is installed may make a second copy for his or her exclusive use on a portable computer.
- **Storage/Network use** You may also store or install a copy of the SOFTWARE PRODUCT on a storage device, such as a network server, used only to run the SOFTWARE PRODUCT on your other COMPUTERS over an internal network; however, you must acquire and dedicate a license for each separate COMPUTER on which the SOFTWARE PRODUCT is run from the storage device. A license for the SOFTWARE PRODUCT may not be shared or used concurrently on different computers.

DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS:

Limitations on Reverse Engineering, Decompilation, and Disassembly. You may not reverse engineer, decompile or disassemble the SOFTWARE PRODUCT, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.

Separation of Components. The SOFTWARE PRODUCT is licensed as a single product. Its component parts may not be separated for use on more than one COMPUTER.

Rental. You may not rent, lease or lend the SOFTWARE PRODUCT.

Support Services. Any supplemental software code provided to you as part of the Support Services shall be considered part of the SOFTWARE PRODUCT and subject to the terms of this EULA.

Software Transfer. The initial licensee of the SOFTWARE PRODUCT may make a one-time permanent transfer of this EULA and SOFTWARE PRODUCT only directly to an end user. This transfer must include all of the SOFTWARE PRODUCT (including all component parts, the media and printed materials, any upgrades, this EULA, and, if applicable, the Certificate of Authenticity). Such transfer may not be by way of consignment or any other indirect transfer. The transferee of such one-time transfer must agree to comply with the terms of this EULA, including the obligation not to further transfer this EULA and SOFTWARE PRODUCT.

Termination. Without prejudice to any other rights, the MANUFACTURER may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all copies of the SOFTWARE PRODUCT and all of its components parts.

UPDATES
If the SOFTWARE PRODUCT is labelled as an update, you must be properly licensed to use a product identified by the MANUFACTURER as being eligible for the update in order to use the SOFTWARE PRODUCT. A SOFTWARE PRODUCT labelled as an update replaces and/or supplements (and may disable) the product that formed the basis for your eligibility for the update. You may use the resulting updated product only in accordance with the terms of this EULA. If the SOFTWARE PRODUCT is an update of a component of a package of software programs that you licensed as a single product, the SOFTWARE PRODUCT may be used and transferred only as part of that single product package.

The SOFTWARE PRODUCT may not be separated for use on more than one computer.

COPYRIGHT
All title and copyrights in and to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text, and "applets" incorporated into the SOFTWARE PRODUCT), the accompanying printed materials, and any copies of the SOFTWARE PRODUCT are owned by the MANUFACTURER. All title and intellectual property rights in and to the content that may be accessed through use of the SOFTWARE PRODUCT is the property of the respective content owner and may be protected by applicable copyright or other intellectual property laws and treaties. This EULA grants you no right to use such content.

You may not copy the printed materials accompanying the SOFTWARE PRODUCT.

DUAL-MEDIA SOFTWARE
You may receive the SOFTWARE PRODUCT in more than one medium. Regardless of the type or size of medium you receive, you may use only one medium that is appropriate for your single COMPUTER. You may not RUN the other medium on another COMPUTER. You may not loan, rent, lease, or otherwise transfer the other medium to another user, except as part of the permanent transfer (as provided above) of the SOFTWARE PRODUCT.

LIMITED WARRANTY
FOR THE LIMITED WARRANTY AND SPECIAL PROVISIONS, PLEASE REFER TO THE REGULATIONS PERTAINING TO YOUR COUNTRY.

WARRANTY AND LIABILITY RESTRICTIONS

We guarantee from date of purchase on for 90 days, that the data carrier of this SOFTWARE PRODUCT is free of material and processing errors. In case, that such errors still occur, please send the data carrier back to the MANUFACTURER or its suppliers; we would replace the data carrier gratis. This amends is your unique amends in the warranty case. It gives you certain rights, and you possibly have other rights prescribed legally, that can vary from jurisdiction to jurisdiction.

The SOFTWARE PRODUCT is made at your disposal based on its current condition. A part of the express warranty mentioned above, there are no other warranties or conditions, neither express nor implicit. This excludes also warranty claims with regards to the commercial quality, saleability and adequacy for a certain purpose, or such, caused by applicable law, statutory provisions, business practice or traffic, however not restricted to these. The entire risk with regards of the results and the performance of the program is up to you. Neither the MANUFACTURER nor its distributors nor suppliers have any liability obligations towards you or any other person or institution for any indirect, incidental, special damages or any consequential damage. This is also valid for damages for lost profit, lost or damaged data or for other commercial or economic losses, even if, the MANUFACTURER has been informed about the possibility of such damages or these were foreseeable, or for claims of third party. In any case, the liability of the MANUFACTURER and the one of its distributors and suppliers is restricted to the sum that you have paid for the SOFTWARE PRODUCT. The liability restrictions defined here are valid independent if, if the assumed or real breach of contract is tangent to basic condition or contract agreement, or is a basic breach of contract. Some states or countries do not allow an exclusion or restriction of liability for consequential damages. Therefore, may be the liability restrictions here-mentioned do not apply to you.

Software Update

To perform a Labelstar Office update, proceed as follows:

1. Open **Labelstar Office**.
2. Click **Help** on the **File** tab, and then click **Check for Updates**.
   The **Update Wizard** opens.
3. Follow the instructions in the wizard.

or visit our Updates website to download the latest program version.

Contacts

Product Website

You may find additional information and the latest program version on or website: www.carl-valentin.de

Email

Technical support: support@carl-valentin.de
Ordering and licensing requests: order@carl-valentin.de
General requests: info@valentin-carl.de

System Requirements

Minimal system requirements

- Microsoft Windows 7 SP1/8.1/10 x86/x64
- .Net Framework 4.6 or later (download from http://www.microsoft.com/net/)
- Microsoft Visual C++ 2010 Redistributable (x86)
- Microsoft Visual C++ 2017 Redistributable (x86) (Download from https://support.microsoft.com/de-de/help/2977003/the-latest-supported-visual-c-downloads)
- Microsoft Access 2013 Runtime (x86)
- Recommended printer drivers: Carl Valentin Printer Drivers Version 2.4.1 or later

Note

Some components, such as .Net Framework for example, are not included in the installation program by default. During installation, the program searches for components, downloads them from the internet where applicable and installs them. If you do not have internet access, you can find the necessary components on the program CD in the *Utilities* folder.

Program Variants

Labelstar Office is available in two versions. In the free LITE version, the software is primarily intended for the design of simple labels. For professional requirements, there is the STANDARD version. This makes a broad selection of formats and variables available so that the requirements of almost any industries can be fulfilled.

	LITE	STANDARD
Texte		
TrueType fonts	✔	✔
Printer fonts		✔
Text formatting (Markup Tags)		✔
Curved text		✔
Barcodes		
1D Barcodes	✔	✔
2D Barcodes		✔
GS1 Barcodes		✔
Post Barcodes		✔
HIBC Barcodes		✔
Images	Limited (BMP only)	More than 90 graphic and vector formats (e.g. TIFF, GIF, JPEG, PNG, WMF, BMP, ICO, ...)
PDF support		✔
Variables		
System variables	Limited (date, time, counter and user input only)	More than 30 variables (e.g. date, time, counter, user input, field link, check digit, If-Then-Else statement, ...)
Printer variables		✔
Databases		✔
Logging		✔
Memory Card Support		✔
Symbols		✔
Label Export		✔
SAPscript ITF print template		
PDF file		
Image file		
Printing		
Internal printer protocoll (CVPL)	Data is transferred graphically	✔ (Carl Valentin Printer Driver Version 2.4.1 or later)
Print preview		✔
Save print settings with label		✔
Column printing		✔
Two-color printing		✔
OLE Automation (LSOffice.dll)		✔

Import Labelstar Plus Labels		✔
Label Converter (batch processing)		✔
Additional Applications		
Quick Print		✔
Print Form		✔
Print Manager		✔
Folder Monitor (automated printing)		✔

Imprint

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

Phone: +49 (0) 7720 9712 - 0
Email: info@carl-valentin.de

Copyright © 2021 Carl Valentin GmbH

Disclaimer

These descriptions however do not constitute any guaranteed characteristics in a legal sense or in the sense of any product liability. The authors reserve the right to make changes to the software, to announce any person without obligation these changes. No warranty is accepted for the correctness of the contents of this document. As errors can never be completely avoided, despite all efforts made, we are grateful for any information regarding errors.

Trademark Notifications

All product names mentioned in this document may be trademarks or registered trademarks of their respective owners.