Etikettenvorlagen

Etikettenvorlagen (Label Templates) für COBI.wms sind HTML-Dateien, die spezielle Platzhalterwerte enthalten können, welche von COBI.wms interpretiert werden. Diese Platzhalter werden über eine einfache „Suchen-und-Ersetzen“-Logik durch die jeweiligen Werte ersetzt, bevor das resultierende HTML zur grafischen Darstellung an das Android-System übergeben wird.

Unterstützte Platzhalterwerte

Die folgenden Abschnitte beschreiben die verschiedenen Arten von Platzhaltern, die in COBI.wms-Etikettenvorlagen unterstützt werden.

Einfache Platzhalter

Die Platzhalter müssen innerhalb des HTML-Dokuments im folgenden Format verwendet werden: `@platzhalterName@` — ein @-Symbol vor und nach dem Namen, ohne Leerzeichen.

Dokumentebene	
`docNumber`	Belegnummer
`docForeignNumber`	BP-Referenz des Belegs
`docDate`	Buchungsdatum
`docDueDate`	Fälligkeitsdatum
`docReference`	Referenz 2 (falls vorhanden)
`docComments`	Bemerkungsfeld
Geschäftspartner des Belegs	
`businessPartnerCode`	Code des Geschäftspartners
`businessPartnerName`	Name des Geschäftspartners
Positionsebene	
`lineNumber`	Positionsnummer **beginnend bei null**
`lineItemName`	Beschreibung (enthält i. d. R. Artikelname)
`lineFreeText`	Freitext
`lineUnitName`	ME-Bezeichnung wie in der Belegposition
`lineBPCatalogCode`	Katalognummer des Geschäftspartners
`lineBPCatalogName`	Katalogbeschreibung des Geschäftspartners
Bezogen auf Mengeneinheit der Position	
`unitCode`	ME-Code (falls nicht „Manual“)
`unitName`	ME-Bezeichnung (ignoriert manuelle Eingaben)
`unitQtyInBaseUnit`	Menge in Basis-ME
Artikelebene	
`itemCode`	Artikelnummer
`itemName`	Artikelname
`itemForeignName`	Fremdbezeichnung / Alt. Artikelnummer
`itemBarcode`	Primärer Barcode (Text)

Dokumentebene	
itemBarcodeGTIN12	Barcode in GTIN-12-Format
itemBarcodeGTIN13	Barcode in GTIN-13-Format
itemBarcodeGTIN14	Barcode in GTIN-14-Format
itemSupplierCatalogNumber	Lieferantenkatalognummer
itemAdditionalIdentifier	Zusatzkennung (DB-Feld SWW)
itemGroupName	Artikelgruppenname
itemPurchasePackQty	Anzahl Artikel pro Einkaufspackung
itemSalesPackQty	Anzahl Artikel pro Verkaufspackung
itemPurchaseUnitQty	Bestand pro Einkaufs-ME
itemSalesUnitQty	Bestand pro Verkaufs-ME
itemPurchaseUnitName	Name der Einkaufs-ME
itemSalesUnitName	Name der Verkaufs-ME
itemStockUnitName	Name der Lager-ME
itemPurchaseLength	Länge der Einkaufs-ME
itemPurchaseWidth	Breite der Einkaufs-ME
itemPurchaseHeight	Höhe der Einkaufs-ME
itemPurchaseLengthUnit	Maßeinheit für Länge Einkauf
itemPurchaseWidthUnit	Maßeinheit für Breite Einkauf
itemPurchaseHeightUnit	Maßeinheit für Höhe Einkauf
itemSalesLength	Länge der Verkaufs-ME
itemSalesWidth	Breite der Verkaufs-ME
itemSalesHeight	Höhe der Verkaufs-ME
itemSalesLengthUnit	Maßeinheit für Länge Verkauf
itemSalesWidthUnit	Maßeinheit für Breite Verkauf
itemSalesHeightUnit	Maßeinheit für Höhe Verkauf
itemPurchaseUnitContents	Inhalt der Einkaufs-ME in Basis-ME
itemSalesUnitContents	Inhalt der Verkaufs-ME in Basis-ME
itemStockUnitContents	Inhalt der Lager-ME in Basis-ME
Chargenebene	
batchNumber	Chargennummer
batchProductionDate	Produktionsdatum
batchProductionGS1	Produktionsdatum im GS1-Format (YYMMDD)
batchExpiryDate	Ablaufdatum
batchExpiryGS1	Ablaufdatum im GS1-Format (YYMMDD)
batchAttr1	Chargenattribut 1
batchAttr2	Chargenattribut 2
batchDetails	Chargenvermerk
Seriennummernebene	
serialNumber	Seriennummer
serialMnfNumber	Herstellerseriennummer
serialLotNumber	Losnummer
serialDetails	Seriennummerdetails
Lagerebene	

Dokumentebene	
`warehouseCode`	Lagercode
`warehouseName`	Lagername
Lagerplatzebene	
`locationCode`	Lagerplatzcode
`locationBarcode`	Lagerplatzbarcode
Kontextabhängig	
`quantity`	Menge der Auswahl (Position/Charge usw.)

Platzhalter auf Dokument- und Positionsebene sind nur gültig, wenn das Etikett aus einer Belegposition heraus gedruckt wird.

Die `itemBarcodeGTIN…`-Platzhalter stellen sicher, dass die Barcode-Länge dem jeweiligen GTIN-Standard entspricht, oder setzen ggf. **Nullen ein**, um weiterhin ein gültiges Format zu erzeugen.

Artikelpreise

Preise können über den Platzhalter `@itemPrice(x)@` aus Preislisten ausgegeben werden.

Format-Beispiele:

- `5.00 USD` (bzw. geräteeinstellungabhängig `5,00 USD`)

Weitere Varianten:

Platzhalter	**Beschreibung**
`itemPriceRawValue(x)`	Unformatiert (z. B. `5.5`)
`itemPriceFormattedValue(x)`	Formatiert nach Gerätesprache
`itemPriceCurrencyCode(x)`	3-stelliger Währungscode
`itemPriceCurrencySymbol(x)`	Währungssymbol (z. B. €)

Barcodegrafiken

Spezial-Platzhalter zur Anzeige tatsächlicher Barcodes:

```
@barcode(FORMAT,BREITE,HÖHE,INHALT)@
```

Unterstützte FORMATE:

- CODEBAR
- CODE_39 / CODE_93 / CODE_128
- DATA_MATRIX
- EAN_8 / EAN_13
- QR_CODE
- UPC_A / UPC_E

Beispiel:

```
@barcode(EAN_13,100,50,@itemBarcodeGTIN13@)@
```

GS1-Unterstützung

GS1-Barcodes erfordern als ersten Charakter: **|**

Beispiel:

```
@barcode(CODE_128,100,75,|01@itemBarcodeGTIN14@10@batchNumber@|)@
```

Präzisere HTML-Kontrolle

Alternativ:

* `@barcodeSrc(…)@` → nur `src`-Attribut * `@barcodeBase64(…)@` → nur Base64-Daten

Benutzer-Eingaben

Eingabeplatzhalter:

```
@input(NAME)@
```

Mit Anzeigename:

```
@input(quantity[Anzahl Kisten])@
```

Mit Auswahlwerten:

```
@input(Currency;EUR,USD,GBP)@
```

Datums-Platzhalter

```
@date(FORMAT)@
```

Beispiele:

`@date(yyyy-MM-dd)@` → `2025-10-24`
`@date(dd.MM.yyyy)@` → `24.10.2025`

Datums-Umformatierung

Der spezielle Platzhalter zur Datumsumformatierung kann verwendet werden, um ein Datum in einem bestimmten Format einzulesen und anschließend in einem anderen Format auszugeben. Die Verwendung sieht wie folgt aus:

```
@dateReformat(DATUM|VON_FORMAT|ZU_FORMAT)@
```

HINWEIS: Keine Leerzeichen vor oder nach den Klammern oder den trennenden senkrechten Strichen setzen! (Leerzeichen würden als Teil des jeweiligen Parameters interpretiert.)

Dieser Platzhalter verwendet das senkrechte Strichsymbol | anstelle von Kommata zur Trennung der Parameter, da die Parameter selbst Kommata enthalten können.

Die Funktionsweise ist wie folgt: Der Text, der im Parameter DATE gefunden wird, wird entsprechend dem Format FROM_FORMAT interpretiert, sodass die Software weiß, welches Datum bzw. welche Uhrzeit gemeint ist, und anschließend im Format TO_FORMAT ausgegeben.

Die Parameter FROM_FORMAT und TO_FORMAT entsprechen dem FORMAT-Parameter des regulären date()-Platzhalters, wie im vorherigen Abschnitt beschrieben.

Im folgenden Beispiel wird das Ablaufdatum einer Charge in einem benutzerdefinierten Format ausgegeben — unter Ausnutzung der Tatsache, dass der Platzhalter @batchExpiryGS1@ immer ein Datum im Format yyMMdd liefert.

Beispiel:

```
@dateReformat(@batchExpiryGS1@|yyMMdd|dd.MM.yyyy)@
```

JavaScript-Integration

Das HTML-Dokument, das nach der Verarbeitung einer COBI.wms-Etikettenvorlage entsteht, wird durch eine vollwertige Browser-Engine dargestellt. Das bedeutet, dass in der Vorlage nicht nur CSS, sondern auch JavaScript verwendet werden kann. Dies muss jedoch explizit aktiviert werden, indem der folgende Pseudo-Platzhalter irgendwo innerhalb der Datei eingefügt wird, z. B. innerhalb eines HTML-Kommentars am Anfang der Datei:

```
@useJavaScript@
```

HINWEIS: Wenn JavaScript aktiviert ist, wird der Druck des gerenderten HTML-Dokuments nicht mehr automatisch ausgelöst. Stattdessen muss der Druck manuell aus dem JavaScript-Code gestartet werden, indem an geeigneter Stelle cobiwms.print() aufgerufen wird.

Während der Ausführung des JavaScript-Codes steht das spezielle Objekt cobiwms zur Verfügung, das eine Reihe von Funktionen bereitstellt. Diese werden im Folgenden erläutert.

```
cobiwms.get(name)
```

Der Parameter name muss ein String sein. Er wird als Platzhalter interpretiert und sein Wert wird als String zurückgegeben. Beispiel: Ein Aufruf von cobiwms.get(“itemName”) liefert den Artikelnamen als Text zurück.

(Man könnte zwar auch Platzhalter wie @itemName@ direkt im Code verwenden, aber dann würde der Wert unverarbeitet in den Code eingefügt werden. Das heißt: Man müsste ihn in Anführungszeichen setzen wie “@itemName@”, und sobald im tatsächlichen Wert selbst Anführungszeichen vorkommen, würde der Code fehlschlagen. Deshalb ist die Verwendung von cobiwms.get() in JavaScript deutlich

sicherer.)

```
cobiwms.prompt(title, callback)
```

Durch den Aufruf dieser Funktion zeigt die App ein Popup mit einem Eingabefeld für den Benutzer an. Der Parameter `title` muss ein String sein und bestimmt die Überschrift des Popups. Der Parameter `callback` muss ebenfalls ein String sein und eine Funktion repräsentieren. Diese Funktion wird nach Bestätigung der Eingabe aufgerufen und erhält dabei einen Parameter, der den vom Benutzer eingegebenen Wert als String enthält.

Zugriff auf Platzhalterwerte:

```
window.handleCurrency = function (currency) {
    ...
}

cobiwms.prompt("Enter currency", "window.handleCurrency")
```

Beispielvorlagen

Zwei vollständige Beispiel-HTML-Dateien:

Datei 1

Einfaches Etikett mit Barcode, Preis & Zeitstempel.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <style>
        @page {
          /* You can use width/height, or a standardized size. */
          /* For example, the following two are equivalent: */
          /* size: 148mm 105mm; */
          /* size: A6 landscape; */
          size: A6 landscape;

          /* Don't change, use the container padding below. */
          margin: 0;
          padding: 0;
        }

        body {
          /* We could use 100vw/100vh to cover the whole size declared with
@page, but this doesn't
             work with ZPL printing, so repeat the page dimensions
explicitly.  Note that we can't
```

```
            use standardized sizes like A6 here.  The height should be
minimally reduced to make
            sure the HTML renderer doesn't start a second page. */
          width: 148mm;
          height: 104mm;

          /* Don't change, use the container below. */
          margin: 0;
          padding: 0;

          /* Useful to diagnose dimension issues. */
          outline: 0.5mm solid black;
          outline-offset: -0.5mm;
        }

        .container {
          /* Don't change. */
          box-sizing: border-box;
          position: relative;
          width: 100%;
          height: 100%;

          font-family: sans-serif;
          font-size: 18pt;

          /* Global padding from the edges. */
          padding: 3mm;
        }
    </style>
</head>
<body>
<div class="container">

    <div style="float: left;">
        <b>Code:</b> @itemCode@
    </div>

    <div style="float: right;">
        <b>Price:</b> @itemPriceCurrencySymbol(1)@
@itemPriceFormattedValue(1)@
    </div>

    <div style="text-align: center; margin-top: 24mm;">
        @itemName@
    </div>
    <div style="text-align: center; margin-top: 2mm;">
        @barcode(CODE_128,150,75,|90@itemCode@|10@batchNumber@)@
    </div>
    <div style="text-align: center; margin-top: 1mm; font-size: 0.8em;">
        (90)@itemCode@(10)@batchNumber@
    </div>
```

```
    <div style="position: absolute; left: 0; bottom: 0; padding: inherit;">
        <b>@date(yyyy-MM-dd HH:mm)@</b>
    </div>

    <div style="position: absolute; right: 0; bottom: 0; padding: inherit;">
        <b>COBI.wms Sample Label</b>
    </div>

</div>
</body>
</html>
```

Datei 2

Beispiel mit Benutzereingaben & JavaScript.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <style>
        @page {
          /* You can use width/height, or a standardized size. */
          /* For example, the following two are equivalent: */
          /* size: 148mm 105mm; */
          /* size: A6 landscape; */
          size: A6 landscape;

          /* Don't change, use the container padding below. */
          margin: 0;
          padding: 0;
        }

        body {
          /* We could use 100vw/100vh to cover the whole size declared with
@page, but this doesn't
             work with ZPL printing, so repeat the page dimensions
explicitly.  Note that we can't
             use standardized sizes like A6 here.  The height should be
minimally reduced to make
             sure the HTML renderer doesn't start a second page. */
          width: 148mm;
          height: 104mm;

          /* Don't change, use the container below. */
          margin: 0;
          padding: 0;

          /* Useful to diagnose dimension issues. */
```

```
          outline: 0.5mm solid black;
          outline-offset: -0.5mm;
        }

        .container {
          /* Don't change. */
          box-sizing: border-box;
          position: relative;
          width: 100%;
          height: 100%;

          font-family: sans-serif;
          font-size: 18pt;

          /* Global padding from the edges. */
          padding: 3mm;
        }
    </style>

    <!-- Input definitions:
    @input(text[Insert value])@
    @input(selection[Select value];Value 1,Value 2,Value 3)@
    @input(codeFormat[Barcode type];CODE_128,DATA_MATRIX,QR_CODE)@
    -->

    <script>
        // @useJavaScript@
        window.onload = function() {
            var insert = document.getElementById('js-insert')
            insert.textContent = "Hello World!"
            cobiwms.print()
        }
    </script>
</head>
<body>
<div class="container">

    <div style="float: left;">
        <b>JS-Insert:</b> <span id="js-insert"></span>
    </div>

    <div style="float: right;">
        <b>Selection:</b> @input(selection)@
    </div>

    <div style="text-align: center; margin-top: 24mm;">
        @itemName@
    </div>
    <div style="text-align: center; margin-top: 2mm;">
        @barcode(@input(codeFormat)@,150,75,|90@itemCode@|10@batchNumber@)@
    </div>
```

```
    <div style="text-align: center; margin-top: 1mm; font-size: 0.8em;">
        (90)@itemCode@(10)@batchNumber@
    </div>

    <div style="position: absolute; left: 0; bottom: 0; padding: inherit;">
        <b>Input:</b> @input(text)@
    </div>

    <div style="position: absolute; right: 0; bottom: 0; padding: inherit;">
        <b>COBI.wms Sample Label</b>
    </div>

</div>
</body>
</html>
```