qrbill v2.00

LaTeX-package to create QR-bills based on the Swiss payments standards

Marei Peischl <marei@peitex.de>

2023/02/28

A collaborative project of peiTeX and foobar LLC

Contents

1. Introduction **2**

2. Example of use **2**

3. Technical Requirements **3**

4. Package options **3**

5. Data setup **5**

6. Parsing dates **6**

7. String replacements **6**

8. Setup for the preconfigured Swiss QR-bill **6**
8.1. Preconfigured data 6
8.2. Address data 7
8.3. Billing information 8
8.4. AV-Parameters 8
8.5. Remaining data elements 8
8.6. The Swiss cross icon 9
8.7. Overview over all data fields 9

9. Setup for EPC QR codes **10**
9.1. Preconfigured data 10
9.2. Required data 11

A. Example of standalone qrbill **12**

1. Introduction

qrbill.sty, the LaTeX-package, is the Free Software and Open Source answer to the launch of the swiss payments conversion, introduced mid 2020, replacing the former payment order slip by a restructured, QR enhanced slip. The published source code is based on the payment standard guidelines issued for Switzerland and intentionally has a modular structure to enable further development and adaptation for international use.

By means of the LaTeX framework, the qrbill can simply be integrated into existing accounting environments and automated billing systems.

2. Example of use

The easies way to use the qrbill package is to create a bill with the standalone documentclass:

```
\documentclass{standalone}
\usepackage[ngerman]{babel}

\usepackage{xcolor}
\usepackage{qrbill}

\begin{document}

\QRbill[
creditor={foobar LLC\\
    Postfach 404\\
    2342 Zurich\\
    CH},
Account=CH1280808005649899718,
% insert additional data here
]

\end{document}
```

`\QRbill [` ⟨*Data Setup*⟩ `]` The macro **\QRbill** creates a box of 210 mm × 105 mm size. It will include a qrbill as described in the design guide [2] and insert all available data.

There is also an example in the appendix showing all data fields (see appendix A on page 12).

The headings are provided in German, English, Italian and French and will be translated using the document's language setup. The terms have been taken from the official guidelines.

3. Technical Requirements

qrbill has been built for TeXLive 2020. Currently it is not possible to ensure it's functionality with older Releases of TeXDistributions.

The packages loaded by qrbill are:

- expl3
- fontspec (except one is using a custom font setup)
- graphicx
- scrbase (which is part of the koma-script bundle)
- qrcode
- iftex
- l3keys2e
- numprint

As a default font "Liberation Sans" will be used. This will be loaded using fontspec, which would require the use of LuaLaTeX or XeLaTeX as a compiler. To have a workaround for this, see the `font` option on page 4 for a custom font setup.

4. Package options

`billinginfo` (true/false) (default: `true`)
Activate/deactivate the automatic fill of the BillingInformation. If this is set to `false` one still can add data to this field using the `BillingInformation` key, as Described in section 5.

`creditorprefix` (String) (default: `CR-/UD-`)
`debtorprefix` For custom setups the predefined prefixes for the debtor and creditoraddress can be changes. The initial configuration is a requirement of the Swiss standards.

`data-to-string` (true/false) (default: `false`)
Converts data to be used to setup the billing QRcode into a string. This option has to be updated if the data should not be treated as ordinary LaTeX code. You can use this to automatically escape special characters.

`font` (Frutiger/Arial/Helvetica/Liberation Sans/custom) (default: `Liberation Sans`)
The official guideline for the Swiss qrbill limits the choice of the font to the first four values of this option. The font selection and sizes currently hardcoded to fit to the size. Custom setups might be provided in the future.

In case one wants to use a custom font, you can set the option `font=custom`. This will disable all font setup and not prevent the fontspec package from being loaded. You can redefine the macro `qrbillfont` to use any font available on your system, for example:

```
\usepackage{fontspec}
% The Laconic font has to be installed
\renewcommand*{\qrbillfont}{\fontspec{Laconic}}
```

`frame` (true/false/top/bottom/none) (default: `true`)
Switch to disable the frame around the created QRbill. The top/bottom options should be prefered when the bill is embedded into a document of a4 papersize. The option none also disables the vertical line inside the bill as this may be included on preprinted paper.

`ibanseparator` (tokenlist) (default: `\,`)
Set the tokenlist to separate the account numbers. See option `sep-iban` for further information.

`icon` (swiss-cross/filename) (default: `<empty>`)
The swiss standard describes the placement of a swiss-cross icon in the center of the QRcode. This can be achieved using this option. The default is set empty, so no icon will be placed. If this option holds another string than “swiss-cross” it will be interpreted as filename and try to load a custom image.

`iconwidth` (length) (default: `7mm`)
This option allows custom scaling for custom icons.

`ignore-if-empty` (comma list) (default: `<empty>`)
Allows to define a list of fields for the qrscheme to be ignored if they have empty values. For the swiss QRbill standard this is recommended for the fields “AV1-Parameters”, “AV2-Parameters” and “BillingInfo”.

`qrmode` (package/lua) (default: `package`)
Selects the mechanism for QRcode generation. With Version 2.0 the package supports usage of the the luaqrcode library [3]. This can be enabled by using the `qmode=lua` option.

`qrscheme` (Name of a custom QRbill scheme) (default: `swiss`)
Loads the definitions for the QRcode and the BillingInformation. Currently only the

swiss qrbill scheme is available, but one can define own variants based on this file. Users can copy the file swiss.qrbill-cfg.tex as an example.

ferenceseparator (tokenlist) (default: `\,`)
Set the tokenlist to separate the reference codes. See option `sep-reference` for further information.

`sep-iban` (integer) (default: `0`)
`sep-reference` Set the size of character groups to separate these. Positive values will be counted left to right. Negative ones the other way round. The most variants of the Swiss QRbill use values of `sep-iban=4` and `sep-reference=-5` e. g.:

Iban: CH12 8080 8005 6498 9971 8
Reference: 00 00000 00000 00000 00000 00152

`separate` (false/text/symbol) (default: `text`)
Setup the separation rule between the document and the qrbill. If the value is text or symbol the Info "Separate before paying in" or the corresponding translation will be printed on top of the frame. This option will be ignored it the frame has no top rule.

`replace-tilde` (true/false) (default: `false`)
`replace-umlauts` Add automatic string replacements for ~ and Umlauts. In general the Data has either to be parsed as string or be expandable. There are some specials about active characters like ~. The option `replace-tilde=true` will replace this by an ordinary space within the data string for the QRcode.

Similarly the replace-umlauts options is mapping Umlauts to the two character representation. This can be used if UTF-8 is not supported. Custom replacements can be set up using `\QRbillAddCustomReplacement` as described in section 7.

`vrule` (true/false/symbol) (default: `true`)
Allows customization of the vertical rule between payment part and receipt. The value `vrule=symbol` acts similar to the `separator=symbol` option.

5. Data setup

`\qrbillsetdata` `*{`⟨*data*⟩`}` The data fields can be filled either using the optional argument of `\QRbill` or using the macro `\qrbillsetdata`. Section 8.7 shows a list of all available fields and appendix A provides an example of use. For further information on the construction of the QRcode see the specification [1].

To simplify the usage qrbill provides an interface to use the different types of address data and is able to create a "billing information" string as described in [1] using the single data elements.

Version 1.04 adds a starred variant to expand the argument before setting the data. This might be usefull for using counters or other variables inside the command.

6. Parsing dates

In version v1.04 qrbill introduced the functionality to parse dates. For the swiss data scheme the fields `invoicedate` and `vatdate` are prefilled with the compilation date.

`\QRbillParseDate {⟨year⟩}{⟨month⟩}{⟨day⟩}` To be compatible with language specific setups the usage of `\today` is not allowed directly but one can use

```
\qrbillsetdata*{
    invoicedate=\QRbillParseDate{\the\year}{\the\month}{\the\day}
}
```

By default this setting will return the ISO date (YYYY-MM-DD) but the swiss scheme is already changing this to the desired (YYMMDD) structure.

7. String replacements

Following a feature request qrbill received support for custom replacements. Those allow the use of special characters within the data fields. The corresponding options `replace-tilde` and `replace-umlauts` provide preset variants for the most common replacements. In case you want to add another you can use the following structure:

```
\AddQRbillCustomReplacement{Ä}{Ae}
```

8. Setup for the preconfigured Swiss QR-bill

8.1. Preconfigured data

The initial configuration which uses the Swiss standard already adds the following data:

```
\SetupQrBill{
  QRType=SPC,
  Version=0200,
  CodingType=1,
  Trailer=EPD,
}
```

These fields are required to use these values for Version 2.2 of the QRbill standard. If any future changes on the standard would require changes, they can be overwritten or redefined by using a custom implementation file, similar to swiss.qrbill-cfg.tex.

Be aware that version 2.1 of the standard requested to use the versioncode `0210`. With the update to version 2.2 of the standard this was changed to `0200`. This change was caused by a discrepancy between the standard and the commong implementation. Most of them continued to use the version code `0200` therefore SIX decided to keep this error and created a workaround with finally documenting this change in version 2.2 of the guildelines.

> Note: In collaboration with representatives of the financial center, SIX has decided that only the version designation "0200" is permitted in master version 02. From master version 03 onwards, depiction of subversions is enabled. [1]

The corresponding issue and discussion concerning this especially for version 2.1 can be found at `https://github.com/peiTeX/qrbill/issues/3`.

Additionally the Currency has been initialized to use "CHF". This also can be overwritten. Currently the Swiss standards only allow the values `CHF` or `EUR`.

8.2. Address data

The Swiss payment standards for QR-bills require the following data for the address of creditor (Prefix `CR-`) and debtor (Prefix `UD-`):

`AddressType`	S	K
`Name`	Name	
`Address1`	Street	Street with number
`Address2`	Number	Postal code with City
`ZIP`	Postal code	Enforced Empty
`City`	City	Enforced Empty
`Country`	Country Code	

To set an address of type "K" one can use they option key `debtor` or `creditor`. Type "S" can be achieved using the starred variant (`debtor*`/`creditor*`).

```
creditor={Name\\
Street Nr.\\
PostalCode City\\
CountryCode},
```

```
creditor*={Name\\
Street\\
Nr.\\
PostalCode\\
City\\
CountryCode},
```

8.3. Billing information

The billing information is a string which can consist of the following data elements:

`invoicenum`	invoice number
`invoicedate`	invoice data
`customerref`	customer reference
`vat`	VAT number
`vatdate`	VAT date
`vatdetails`	VAT percentage / details
`importvat`	import VAT
`conditions`	payment conditions

The data has to be in the required format. qrbill might add features to automatically validate this in the future.

Beside setting the data fields all on their own, qrbill can also handle a complete billing information string. One could simply set it by

```
BillingInfo=//S1/10/10201409/11/190512/20/1400.000-53/30/106017086
  /31/180508/32/7.7/40/2:10;0:30
```

8.4. AV-Parameters

The QRcode can use 2 AV Parameters. They have to carry the Prefix of “Name AV1:”/“Name AV2:”. This will be automatically added if the keys `AV1`/`AV2` are used instead of `AV1-Parameters`/`AV2-Parameters`.

8.5. Remaining data elements

The remaining data elements should be set directly. For restrictions on the content see the specification. Currently there is no validation taking place. The remaining for the preconfigures Swiss standard are: Account, Amount, ReferenceType, Reference, Message,

8.6. The Swiss cross icon

[2] reguires a swiss cross of 7 mm size to be placed in the center of the QRcode. This package supports this by using the package option for the icon:

```
\usepackage[icon=swiss-cross, …]{qrbill}
```

This setting is not activated by default.

8.7. Overview over all data fields

Data fields which are used in the QRcode:

QRType
Version
CodingType
Account
CR-AddressType
CR-Name
CR-Address1
CR-Address2
CR-PostalCode
CR-City
CR-Country
UCR-AddressType
UCR-Name
UCR-Address1
UCR-Address2
UCR-ZIP
UCR-City
UCR-Country
Amount
Currency
UD-AddressType
UD-Name
UD-Address1
UD-Address2
UD-PostalCode
UD-City
UD-Country
ReferenceType
Reference
Message
Trailer

BillingInfo
AV1-Parameters
AV2-Parameters

Data fields, which are custom created by the qrbill package to simplify the use:

creditor and creditor*
debtor and debtor*
AV1 and AV2

Data fields to automatically create the BillingInfo string. The key in parentheses indicates the prefix to be used to construct the string.

//S1 (prefix)
invoicenum (10)
invoicedate (11)
customerref (20)
vat (30)
vatdate (31)
vatdetails (32)
importvat (33)
conditions (40)

9. Setup for EPC QR codes

EPC QR codes can be used for SEPA credit transfer (SCT). They do not include any data about the debtor or tax information. Therefore they may only be used with the starred variant of `\QRbill*` or the user has to provide additional data interfaces.

9.1. Preconfigured data

The epc qrscheme provided by this package preset the following data:

```
\SetupQrBill{
      QRType=BCD,
      Version=002,
      CodingType=1,% UTF-8, 2 would be ISO
      Trailer=SCT,
    }
```

These fields are required to use these values for the current version of the EPC QR code.

9.2. Required data

The data fields required by the EPC QRcode can be set directly. A full example for an easy EPC qrcode would be

```
\QRbill*[
  BIC=BYLADEM1GLS,
  Account=DE68430609671013251700,
  Name=peiTeX,
  Amount=EUR123.42,
  Message={Invoice 2022:1337, customer 1337}
]
```

Change History

v1.00
General: First official version 1
v1.01
General: Add separate option to modify the top rule of the QR bill 5
v1.02
General: add ibanseparator option . . . 4
add referenceseparator option 5
add sep-iban/sep-reference option . . 5
Fix placement issue with separate option 5
Note the version number inconsistency in the official guidelines 7
v1.03
General: set sep-iban and sep-reference in examples 12
Update to swiss qrbill standard version 2.2 6
v1.04
General: Add basic support for EPC QR codes 10
Add date parsing mechanisms 6
Add starrred variant of qrbillsetdata 5
v1.05
General: add data-to-string option . . . 3
Add replace-tilde/replace-umlauts options 5
Add support for custom string replacements 6
v1.06
General: Add ignore-if-empty option . . 4
symbol option. 5
v2.00
General: qrencode.lua now is part of the qrbill package 4

References

[1] Swiss Implementation Guidelines QR-bill: Technical and professional specifications of the payment part with Swiss QR Code and of the receipt. Version 2.2, with effect from 22 February 2021. https://www.paymentstandards.ch/dam/downloads/ig-qr-bill-en.pdf. Last checked 2022-08-28

[2] Style Guide QR-billStyle Guide QR-bill: The right layout pays off. Layout rules and recommendations for the payment part with Swiss QR Code and for the receipt. https://www.paymentstandards.ch/dam/downloads/style-guide-en.pdf

[3] Luaqrcode – Pure Lua QR Code library. By Patrick Gundlach (SPEEDATA GMBH) and contributors. http://speedata.github.io/luaqrcode/. Last checked 2022-08-28

A. Example of standalone qrbill

```
% !TeX program=lualatex
\documentclass{standalone}
\usepackage[nswissgerman]{babel}

\RequirePackage{xcolor}
\usepackage[
  icon=swiss-cross,
  separate=false,
  sep-iban=4,
  sep-reference=-5
]{qrbill}

\begin{document}

\QRbill[
  creditor*={foobar LLC\\
Postfach\\
404\\
2342\\
Zurich\\
CH},
  Account=CH1280808005649899718,
  vat=123123123,% VAT number with stripped CH and periods
  debtor*={peiTeX\\
TeXnikerweg\\
78\\
23420\\
Hamburg\\
DE},
  Amount=1337.42,
  Message=Bestellung vom 27.06.2020,
  invoicenum=100-4242,
% invoicedate=200701,%yymmdd, preset to todays values
  vatdetails=0,% 0% VAT
% invoicedate=200701,%yymmdd, preset to todays values
  AV1=LX;F00BAR;2342,
]
```

```
\end{document}
```

B. qrbill scrletter example

```
% !TeX Program=lualatex
\documentclass[foldmarks=b]{scrletter}
\usepackage[
  frame=top,
  sep-iban=4,
  sep-reference=-5,
]{qrbill}

% Setup layer:
% This requires the scrlayer package which is already loaded by scrletter
\DeclareNewLayer[
align=bl,
voffset=\paperheight,
hoffset=0pt,
contents={\QRbill},
width=\paperwidth,
height=105mm,
]{qrbill}



\begin{document}

\qrbillsetdata{
  creditor*={foobar LLC\\
Postfach\\
404\\
2342\\
Zurich\\
CH},
  Account=CH1280808005649899718,
  vat=123123123,% VAT number with stripped CH and periods
  debtor*={peiTeX\\
TeXnikerweg\\
78\\
23420\\
Hamburg\\
DE},
  Amount=1337.42,
  Message=Bestellung vom 27.06.2020,
  invoicenum=100-4242,
%  invoicedate=200701,%yymmdd, preset to todays values
  vatdetails=0,% 0% VAT
%  vatdate=200701,%yymmdd, preset to todays values
```

```
  AV1=LX;FOOBAR;2342,
}

\setkomavar{fromaddress}{\insertcreditor}

\begin{letter}{\insertdebtor}

\opening{opening}

Text

\closing{closing}

% Add qbill on last page
% For more details
\AddLayersToPageStyle{@everystyle@}{qrbill}

\end{letter}
\end{document}
```