# The zugferd package[*]

Creating electronic and hybrid invoices using LaTeX

Marei Peischl
marei@peitex.de

October 23, 2024

**Abstract**

Invoicing is getting more and more automated. Starting with public sector, within Germany there already is a requirement to stick to the Faktur-X Standard. First Invoices based on this implementation here have been created back in 2021. And this is now the trial to create a more universal and public package to support the current Version of ZUGFeRD and therefore also X-Rechung and Faktur-X.
The fundamental idea of this package was to use the calculation within LaTeX as well. So it also creates the XML filefor the attachment on the fly. To match typical setups there is a wrapper package which usually would also hold the personal Invoicing layout configuration.

## Sponsors & Supporters

Most of this package has been created within my free time and for my personal use. At start, it was not a paid project at all. Since it is addressing business users it would be great if we could keep this actively maintained. If you are able to support this either financially for the maintenance effort, a custom extension, I'd love to hear from you.

Tthis project was financially supported by:

- Pengutronix e.K., https://pengutronix.de
  Special thanks to them, as they also sponsored the minimal portable TeX Live setup.
- Frederik Schwan (Schwan.IT)

[*] This document corresponds to zugferd v0.9, dated 2024-10-23.

# Contents

# 1 Quick start

This package is still in development and does not provide any validation. To ensure your invoice is created correctly you should also validate the output files. There are tools like the [7] providing an easy-to-use interface for the validation. In the appendix I will add some notes on my setup and how I use it within pipelines.

The Bundle provides an example file called `DEMO-rechnung-zugferd.tex`. This includes a basic setup for a valid X-Rechnung currently matching Version 3.0.1 of the standard. Details on the requirements can be found in the documentation at [1].

## 1.1 Disclaimer concerning the zugferd-invoice Package

The included package zugferd-invoice is an example project which might match your own invoicing structure. It holds all the layout information which is static across all the invoices. This package is an example implementation and should not be used in production. It is published as a part of the documentation.

The idea is to create your own version of this package to use your own layout and internally load the zugferd package that way. Of course, it's possible to use a copy of this package within your personal setup. But the syntax used in the DEMO file may change, so you have to ensure yourself to be compatible with updates.

The interfaces for zugferd will hopefully stay the same. At least changes will be announced and build compatible during a deprecation period.

# 2 Package Options

The package supports a few fundamental settings. These have to be set when the package is loaded as they are used internally to setup the scheme or activate the XML mechanism.

`format=` (`xrechnung/xrechnung3.0/xrechnung2.3/basic`) (default: `xrechunng`)

`format` selects the scheme to be used for the zugferd invoice. Currently xrechnung3.0, xrechnung2.3 and the basic scheme are supported.

The value `xrechnung` is set as an alias to `xrechnung3.0` and will always use the latest version supported by zugferd.

`zugferd=` ⟨*boolean*⟩ (default: `true`)

This option can be used to deactivate the XML embedding. It would also disable the the `write-xml` option. This can be used to create a package which can use the same structure to also create invoices without XML attachment. It can also be used with older LaTeX releases than this package requires. There will be a warning, but the visible part should be okay.

`write-xml=` ⟨*boolean*⟩ (default: `true`)

Disable the XML output. This can be used if you want to create the XML attachment with different software than this package.

In that case you can either rename your file to ⟨`\jobname_`*zugferd.xml*⟩ or also adjust the `xml-file` option.

`xml-file=` ⟨*filename*⟩ (default: `\jobname_zugferd.xml`)

Adjust the file name of the created or loaded XML file.

The option `xrechnung` is only used internally to set the global parameters for all `xrechnung` variants.

`auto-exemption=` (⟨*boolean*⟩) (default: `true`)

zugferd tries to automatically add an exemption-reason for the most common VAT categories. In case a more specific reason is required this setting can be disabled and everything should be configured manually. See subsubsection 5.3.2 for more explanation of this feature and the categories this applies to.

## 3 User Commands

The end user is only asked to set or access the data to be used by zugferd.

```
\SetZUGFeRDData*{⟨key value list⟩}
```

`\SetZUGFeRDData` `\SetZugferdData` The two modes of `\SetZUGFeRDData` control if the argument is expanded before the fields are set. Depending on the source of the data this might be necessary. Fields which are involved in the calculation will be expanded anyway, but the text fields will not, to support special characters.

`\InsertZUGFeRDData` `\InsertZugferdData`

```
\InsertZugferdData[⟨special mode option⟩]{⟨data-selection⟩}
```

ZUGFerd uses the same data as the XML fileinside the PDF. To simplify the reuse of data this command is designed to simplify the access to data fields, for example:

```
\InsertZUGFeRDData{id}
{\InsertZUGFeRDData[set-today]{date}\today}
\InsertZUGFeRDData[AddressData]{seller}
```

As special modes the command currently supports the following:

By default zugferd tries to find the variable holding the data itself. First a token list is tried, afterwards a string. Global variables are prefered over local ones.

As the variable names may container underscores and the option usually prefers dashes, dashes are converted to underscores for the detection.

`AddressData` Allows `seller` or `buyer` for the data selection. Will print the address, to be used in letters.

`set-today` For dates there also exists the variant which will not print the variable but parse the variable to be used as `\today`. Using this the date format can be controlled easier using the language setting of the document. Here you should take care to use it within a group to restore the real value of `\today` afterwards.

## 4 Commands for template authors

`ZUGFeRD` (*env.*) To simplify the structure of the wrapper package, zugferd provides an environment for the XML mechanism and does the attachment to the PDF file (of course only, if enabled, see section 2). This provides the public interface bundling some steps together to reduce maintenance effort for any template maintainer using this package. It also avoids the use of internal commands.

This environment opens the XML file using `\startWritingZUGFeRDxml` and afterwards writes the XML header including the File and Scheme information, the `ExchangedDocumentContext` and information of the `ExchangedDocument`. Notes will also be written within this step. Afterwards the environment should include all the mechanisms to write the invoice positions as well as summation.

At the end of the environment the footer is inserted, before the output stream is closed using `\stopWritingZUGFeRDxml`. Which also attaches the XML fileto the PDF.

`\startWritingZUGFeRDxml`

`\startWritingZUGFeRDxml` is opening the output stream for the XML file. It also adjusts the indentation. If `write-xml` is false, this option only opens a group to achieve the same structure in both modes.

`\stopWritingZUGFeRDxml`

Here the output stream is closed and the XML fileis attached. In case `write-xml` is not active, the attachment will be made if that's not deactivated separately using `zugferd`. It also ends the group started by `\startWritingZUGFeRDxml`.

## 4.1 Interfaces to write the XML contents

In case you are using `write-xml=true` (which is the default) You need to ensure to call the XML writing functions in the correct order. For example after setting the global invoice data, like it's done in the example file. The minimal example below would create a valid XML. The interface commands are described afterwards.

```
\begin{ZUGFeRD}
  \zugferd_write_Item:nnnnnn {1} {} {Plushie~\TeX\ lion} {31.98} {2}
  ↪  {63.78}
  \zugferd_startInvoiceSums:
  \zugferd_write_TaxEntry:nnnn {S} {19} {63.78} {12.12}
  \zugferd_write_Summation:nnnnnnnn {63.78} {0} {0} {63.78} {12.12}
  ↪  {75.90} {0} {75.90}
  \zugferd_stopInvoiceSums:
\end{ZUGFeRD}
```

`\zugferd_write_Item:nnnnnn`

This command is the interface to write invoice items to the XML file. If the XML interface is enabled this is a reference to the internal command `\__zugferd_insert_TradeLineItem:nnnnnn`.

```
\zugferd_write_Item:nnnnn
{⟨LineID⟩}{⟨optional: item id ("SellerAssignedID")⟩}{⟨item name⟩}
{⟨NetPriceProductTradePrice⟩}
{⟨BilledQuantity⟩}
{⟨LineTotalAmount⟩}
```

Within the product name macros are disabled using `\zugferd_disable_macros:`, see subsection 4.3.

This command is using the local values of tax information as well as the unit code. If you want to overwrite them, adjust them locally using the corresponding options, e.g.:

```
\group_begin:
  \keys_set:nn {zugferd/item}{tax/rate=19, tax/category=S}
  \zugferd_write_Item:nnnnnn {1} {} {Plushie \TeX\ lion} {31.98} {2}
  ↪  {63.78}
  % Code using the data for visual representation
\group_end:
```

This will set the tax rate to 19 % unregarding the global setting.

This structure might look a bit overcomplicated as one might think the options could also be set as an additional argument. This works as long as the Code for the visual part of the invoice is not referencing the internal data. In case you don't do this it's also possible to use the following variant:

`\zugferd_write_Item:nnnnnnn`
`\zugferd_write_Item:ennnnnn`

This is grouping the command and adding an argument in front to add additional options.

```
\zugferd_write_Item:nnnnn
{⟨additional local options⟩}
{⟨LineID⟩}{⟨optional: item id ("SellerAssignedID")⟩}{⟨item name⟩}
{⟨NetPriceProductTradePrice⟩}
{⟨BilledQuantity⟩}
{⟨LineTotalAmount⟩}
```

The example above could then be replaced by

```
\zugferd_write_Item:nnnnnnn {tax/rate=19,tax/category=S} {1} {Plushie-01}
↪  {Plushie \TeX\ lion} {31.98} {2} {63.78}
 % visual representation now may not refer to the data
```

`\zugferd_startInvoiceSums:`
`\zugferd_stopInvoiceSums:`

There is some global data which is placed in the XML fileafter the invoice items have been placed. Typically, in LaTeX this block is started after the items have been printed and will enclose the summation block.

The starting includes the so called "ApplicableHeaderTradeAgreement" which contains the address data of both trade parties, see subsection 5.2 And this will also print the "SpecifiedTradeSettlementPaymentMeans", see subsubsection 5.2.2.

`\zugferd_write_TaxEntry:nnnn`

This command is writing the sum over a tax rate. This command has to be used once per rate applied to the items.

```
\zugferd_write_TaxEntry:nnnn {⟨tax category code⟩} {⟨tax rate in %⟩} {⟨basis
amount the tax applies to⟩} {⟨tax amount⟩}
```

The tax amount could of course be calculated internally. In the example package this is done automatically, but the interface needs to support manual input as a lot of use cases for LaTeX invoicing use it only to create the output file.

`\zugferd_write_Summation:nnnnnnnn`

The total values are all collected with a single macro.

```
\zugferd_write_Summation:nnnnnnnn
{⟨LineTotalAmount⟩}{⟨ChargeTotalAmount⟩}{⟨AllowanceTotalAmount⟩}
{⟨TaxBasisTotalAmount⟩}{⟨TaxTotalAmount⟩}
{⟨GrandTotalAmount⟩}{⟨TotalPrepaidAmount⟩}{⟨DuePayableAmount⟩}
```

This commnd is also writing the payment terms to the XML file. Please be aware that it's in general not possible to calculate the tax values in here, as there might be multiple tax rates applied. This is only taking the sums over all tax entries.

In case you are using some specials like category "E" the exemption reason will also be written at that point. For that it is referencing the current value of the setting.

### 4.2 Commands to temporary disable/re-enable the XML writing interfaces

`\zugferd_enable_XML_interfaces:`
`\zugferd_disable_XML_interfaces:`

As there are a lot of usecases where code is processed multiple times, it's necessary to provide an interface to temporary disable the XML writing mechanism. A lot of these situations appear within table structures whereas a local adjustment would not be helpful. Therefore these adjustments have to be done globally.

The example package zugferd-invoice provides an example for this to ensure the XML data is not written multiple times. The `ZUGFeRD` environment has been constructed that way, that it would automatically enable the interface when it begins and also when it ends, to write the data. So you should ensure this environment is only processed once or use the lower level interfaces directly. Setting up the catcodes to simplify the XML indentation.

### 4.3 Escaping macros inside XML data

`\zugferd_disable_macros:` Since we allow the use of LaTeX code in some fields there has to be a mechanism to disable macros inside the XML output. The mechanism is created similar to the one by hyperref, and we also use some definitions from there to use those as a starting point. To have a detailed list of the redefinition, please have a look at the implementation of this command.

There exists a hook to extend or overwrite these definitions `zugferd/disable-macros`. You can add own redefinitions using this. For example if you want to overwrite the setting mapping a `\newline` to a new line char instead of space, you could add the following to your setup:

```
\hook_gput_code:nnn {zugferd/disable-macros}
    {newline-to-LF}
    {\def\newline{\iow_newline:}}
```

## 5 Adding data to the XML

All data which does not directly depend on amounts or specific items is provided using a key-value interface. For some fields there is the option to define a global preset but locally overwrite it for a specific item. This only applies to data fields used by the writing interfaces described in subsection 4.1.

This package is using the UN/CEFACT Cross Industry Invoice Syntax for the data. Currently it is not planned to implement the UBL syntax as well, but generally this would be possible.

Please be aware that the zugferd package does currently not handle any replacements concerning the content. Therefore it might be necessary to escape special characters, like `&` to `&amp;`. This also applies to `<`, `>`, `"` and `'`. It's technically possible to do this either via active characters or string replacements. But since it's adjusting the content this feature would never be enabled by default. The corresponding issue can be found at `https://github.com/TeXhackse/LaTeX-ZUGFeRD/issues/9`.

In most cases this functionality will be used to change the tax setting or unit for a single item. subsection 4.1 also provided an example for this.

This section will now take all data which can be set using `\SetZUGFeRDData`.

## 5.1 General Invoicing Data

Some of the general data currently supports only one value, which is alreay selected by default. The interface already exists and may be extended later.

`document-type=` (`commercial-invoice`) (default: `commercial-invoice`)

Select the document type. The only supported value currently is `commercial-invoice`. This will select the corresponding type code, which is 380.

### 5.1.1 Invoice number/document ID

`id=` (`komavar`/⟨*document ID/invoice number*⟩) ⟨*initially unset*⟩

This has to be set. Leaving it empty will lead to an invalid XML file.

The value `komavar` would reference the data provided the KOMA-Script letter variabe `invoice`. In case you don't use `scrletter` you should not use this setting. More information can be found in the documentation [4].

### 5.1.2 Currency

`currency=` (`EUR`/`USD`/`CHF`/`€`) (default: `EUR`)

Currently `zugferd` only supports one currency for an invoice. This might be extended later. The currency is pre-configured to use Euro.

### 5.1.3 Dates

`date=` (`auto`/⟨*date formatted as YYYYMMDD*⟩) (default: `auto`)

`delivery-date=` (`auto`/⟨*date formatted as YYYYMMDD*⟩) (default: `auto`)

`due-date=` ⟨*date formatted as YYYYMMDD*⟩ ⟨*initially unset*⟩

Currently there are three kinds of dates implemented. The XML-Standard requires them to use the structure ⟨*YYYYMMDD*⟩. For the day this document was compiled this would be: "20241023" (October 23, 2024).

Instead of providing a date value directly it's also possible to use `\today`. This is done using the  which is the default setting for `date` and `delivery-date`. Please be aware, that this would change if you rebuild the document later. So you might want to use an actual value here.

### 5.1.4 Payment terms

`payment-terms=` (⟨*string*⟩) ⟨*initially unset*⟩

One option to set payment terms is the `due-date` mentioned before. If this is not set or the setting is more complex one can use `payment-terms` to add more information.

This setting is a string. In case there is expansion required this has to be done before.

### 5.1.5 Notes: Adding additional information

`subject=` (`komavar/`⟨*Tokenlist*⟩) ⟨*initially unset*⟩

`fromaddress=` (`komavar/`⟨*Tokenlist*⟩) ⟨*initially unset*⟩

`add-note=` ⟨*Tokenlist*⟩ ⟨*initially unset*⟩

The ZUGFeRD example files[10] use all visible data to add them to the XML as a note. `subject` and `fromaddress` are used to support this. The data should not be too relevant but zugferd want's to support adding additional data to the XML using the note element. So these fields can be left out but in case they are not empty, they will also be written to the XML.

The  corresponds to the mechanism provided by scrletter. It accesses the variable expands it to be used directly. If you don't use this package, you can ignore this setting or add content manually.

## 5.2 Trade parties

The XML scheme knows 6 different Trade Parties:

- Seller
- Buyer
- Payee
- ShipTo
- SellerTaxRepresentative

Currently zugferd supports only Buyer, Seller and ShipTo, but can be easily extended to support the others as well. The data for each party follows the same structure, except the "BuyerReference" which is described later in this section.

Some of the data is optional for specific parties. As this also depends on the selected scheme and version we will not list the details. All fields for a trade party can be set using the "group" named by the party. For example setting all the seller data is done in the following listing:

```
\SetZUGFeRDData{
  seller/name = {peiTeX (Marei Peischl)},
  seller/email = {invoicing@peitex.de},
  seller/vatid = {DE123456789},
  seller/contact= {Marei\\+4900000000\\marei@peitex.de},
  seller/address = {Address Line 1\\Address Line 2},
  seller/postcode = {20253},
  seller/city ={Hamburg},
  seller/country = {DE},
}
```

All this data is saved within a property list, which is internally called `\g__zugferd_`⟨*seller/buyer/shipto*⟩`_prop`. By default this property list is empty. The users themselves have to ensure to add the required data.

The outer braces are not required, if the data does not container an equal sign or a comma. In case the final data is unknown, it's recommended to use them anyway.

⟨*party*⟩/name= ⟨*name*⟩ ⟨*initially unset*⟩

⟨*party*⟩/email= ⟨*email address*⟩ ⟨*initially unset*⟩

⟨*party*⟩/vatid= ⟨*VAT ID*⟩ ⟨*initially unset*⟩

⟨*party*⟩/address= ⟨*address*⟩ ⟨*initially unset*⟩

As shown in the example `address` can use two lines separted by `\\`. It's possble to set all fields for all trade contacts, but e. g. for the `shipto`-party email and vatid will not be used in the XML.

Alternatively it's also possible to use ⟨*party*⟩`/lineone` and ⟨*party/linetwo*⟩ separately. This may be helpful if you use a custom input format. In any way you should ensure that all macros used within the data either are expandable or disabled using `\zugferd_disable_macros:`.

⟨*party*⟩/postcode= ⟨*postal code*⟩ ⟨*initially unset*⟩

⟨*party*⟩/city= ⟨*city*⟩ ⟨*initially unset*⟩

⟨*party*⟩/country= ⟨*country code*⟩ ⟨*initially unset*⟩

The two letter country codes allowed here can be found in [2].

⟨*party*⟩/contact= ⟨*Combined contact data*⟩ ⟨*initially unset*⟩

The contact person can either be set using the combined structure similar to ⟨*party*⟩`/address`. It either consists of 3 or 4 entries, depending on if a department should be used or not.

```
\SetZUGFeRDData{
  seller/contact = {
     ⟨contact-name⟩\\
     ⟨contact-phone⟩\\
     ⟨contact-email⟩
  },
  seller/contact = {
     ⟨contact-name⟩\\
     ⟨contact-department⟩\\
     ⟨contact-phone⟩\\
     ⟨contact-email⟩
  }
}
```

As for `seller/address` it's also possible to set the keys directly:

```
\SetZUGFeRDData{
  seller/contact-name= {⟨contact-name⟩},
  seller/contact-department = {⟨contact-department⟩},
  seller/cotact-phone ={⟨contact-phone⟩},
  seller/contact-email= {⟨contact-email⟩}
}
```

### 5.2.1 Buyer Reference

`buyer/reference=` (`komavar`/⟨*Reference*⟩) ⟨*initially unset*⟩

The reference field only exists for the `buyer` trade party. Depending on the process it's required to use some unique identifier referring to the `buyer`. Within Germany these numbers are called "Leitweg-ID"[6].

In any way the `buyer` may choose what is used here. Also may be some PO number or similar reference.

As defined for other variables the `reference` can also use the  value to refer to the value of komavar `yourref`[4].

### 5.2.2 Payment Means

The payment means are selected by numeric codes. Currently we support:

- 1 = Instrument not defined
- 10 = In cash
- 30 = Credit Transfer
- 31 = Debit Transfer
- 42 = Payment to bank account
- 48 = Bank card
- 49 = Direct Debit
- 57 = Standing agreement
- 58 = SEPA credit transfer
- 59 = SEPA direct debit
- 97 = Clearing between partners

Others may be added in the future but it's not planned to include a full list.

The codes will automatically add the corresponding string inside the "Information" field. The initial version only included German strings, but currently they are also included in English. It's possible to overwrite them using the same structure:

```
\setupZUGFeRDStrings{payment-means}{
  10 = Bargeld,
  58 = Zahlung per SEPA Überweisung.,
}
```

The language selection is done using at hook executed at `\begin{document}` and will try to use the document's language. If this is not defined English will be used.

Internally the commands are predefined as a key-value list like the argument in the example above. They macros are called `\zugferd@paymentMeans@`⟨*languagename*⟩. Currently `zugferd` defines these for `english` and `german` (also `ngerman` as an compatibility alias).

## 5.3 Variables which may be changed per invoice item

Some settings may have the same value for all invoice items. These are defined to take some preset but are set locally. So it's possible to adjust them for a single invoice item if necessary. An example is shown in subsection 4.1.

### 5.3.1 Units

`unit=` (`hour/day/one/piece/`⟨*unit code*⟩) ⟨*initially unset*⟩

The Faktur-X standard requires the unit to be selected. These are called "/UN/CEFACT Common Codes" and can be found withtin [8].

Currently `zugferd` supports `hour` (`HUR`), `day` (`DAY`), `one` (C62) and `piece` (H87). For these the corresponding codes have been implemented within the package. Other units can be selected using the codes listed in [8].

This option is not case sensitive The value is automatically converted to uppercase. If the selected option is different from the predefined ones, there will be a warning, as `zugferd` does not know if the selection is valid or not.

### 5.3.2 Tax category and rate

`tax/category=` ⟨*category code/alias*⟩ (default: `standard`)

The Tax data requires a category code. For details have a look at the Specification [e. g. at 5]. `zugferd` implements all of those, but the user has to take care to select the correct one for each invoice item. The example file includes 2 different VAT values using the same category.

The labels have been chosen to simplify the usage. It's also possible to enter the codes directly. This option is not case sensitive.

`standard` Standard rate and reduced rate item, `category=S`

`zero` Zero rated sale, `category=Z`

`exempt` Exempted from VAT. This requires a reason via `exemption-reason`,`category=E`

`reverse-charge` Reverse Charge, `category=AE`

`intra-community` or `EEA` Intra-Community Supply, `category=K`

`export` Free export item, tax not charged, `category=G`

`O` Services outside scope of tax

`canary-islands` Canary Islands general indirect tax, `category=L`

`ceuta` or `melilla` Ceuta and Melilla, `category=M`

`tax/exemption-reason=` ⟨*Text*⟩ ⟨*initially unset*⟩

`tax/exemption-reason-code=` ⟨*exemption reason code*⟩ ⟨*initially unset*⟩

Add Reasons for a tax exempt, as required by `category=E,K,AE,G,O`. This can either be added using a text (`exemption-reason`) or a predefined code (`exemption-reason-code`). The codes are listed at [9].

In most common cases `zugferd` tries to automatically match them if the package option `auto-exemption` is enabled, which is the default. In that case the following settings would apply:

`S` Exemption reason: ⟨*empty*⟩; Exemption reason code: ⟨*empty*⟩

`Z` Not configured.

`E` Not configured, as there are too many options.

`AE` Exemption reason: Reverse Charge; Exemption reason code: vatex-eu-ae

`K` Exemption reason: Intra-Community Supply; Exemption reason code: vatex-eu-ic

`G` Exemption reason: Export outside the EU; Exemption reason code: vatex-eu-g

`O` Exemption reason: No subject to VAT; Exemption reason code: vatex-eu-o

In case there is no pre-configured selection `zugferd` will create a warning to remind the user to add a selection themselves.

`tax/rate=` ⟨*floating point*⟩ (default: `19`)

The value given will be used for tax calculation. By default it's configured to `19` to match the German standard VAT rate.

`item/start-date=` ⟨*date formatted as YYYYMMDD*⟩ ⟨*initially unset*⟩

`item/end-date=` ⟨*date formatted as YYYYMMDD*⟩ ⟨*initially unset*⟩

With version 0.00 support for `BillingSpecifiedPeriod` was added. This supports setting `start-date` and `end-date` per item. As this is optional, there is no default. The element will be printed if both dates are set, as setting a single one will enforce the element to be invalid. This element should be set as all the other dates (see subsubsection 5.1.3).

# 6 Implementation

`\l__zugferd_tmp_tl`
`\g__zugferd_format_str`
`\g__zugferd_businessProcessId_str`
`\g__zugferd_writeTradeContact_bool`
`\g__zugferd_writePaymentMeans_bool`
`\g__zugferd_conformance_level_str`

```
\tl_new:N \l__zugferd_tmp_tl
\str_new:N \g__zugferd_format_str
\str_new:N \g__zugferd_businessProcessId_str
\bool_new:N \g__zugferd_writeTradeContact_bool
\bool_new:N \g__zugferd_writePaymentMeans_bool
\str_new:N \g__zugferd_conformance_level_str
```

(*End of definition for* `\l__zugferd_tmp_tl` *and others.*)

`format`
`xrechnung`
`write-xml`
`zugferd`
`xml-file`
`auto-exemption`

```
\char_set_catcode_other:N \#%
\keys_define:nn {zugferd} {
 xrechnung .code:n = {
  \bool_gset_true:N \g__zugferd_writeTradeContact_bool
  \bool_gset_true:N \g__zugferd_writePaymentMeans_bool
  \str_gset:Nn \g__zugferd_conformance_level_str {XRECHNUNG}
 },
 format .choice:,
 format / xrechnung3.0 .code:n = {
  \str_gset:Nx \g__zugferd_format_str {
   urn:cen.eu:en16931:2017#compliant#urn:xeinkauf.de:kosit:xrechnung_3.0
  }
  \str_gset:Nx \g__zugferd_businessProcessId_str {
   urn:fdc:peppol.eu:2017:poacc:billing:01:1.0
  }
  \keys_set:nn {zugferd}{xrechnung}
 },
 format / xrechnung2.3 .code:n = {
  \str_gset:Nx \g__zugferd_format_str {
   urn:cen.eu:en16931:2017#compliant#urn:xoev-de:kosit:standard:xrechnung_2.3
  }
  \keys_set:nn {zugferd}{xrechnung}
 },
 format  / basic .code:n = {
  \str_gset:Nx \g__zugferd_format_str {
   urn:cen.eu:en16931:2017#compliant#urn:factur-x.eu:1p0:basic
  }
  \bool_gset_false:N \g__zugferd_writeTradeContact_bool
  \bool_gset_false:N \g__zugferd_writePaymentMeans_bool
  \str_gset:Nn \g__zugferd_conformance_level_str {BASIC}
 },
 format / xrechnung .meta:n = { format = xrechnung3.0 },
 format .initial:n = xrechnung,
 format .usage:n = load,
 write-xml .bool_gset:N = \g__zugferd_write_xml_bool,
 write-xml .initial:n = true,
 write-xml .usage:n = load,
 zugferd .bool_gset:N = \g__zugferd_active_bool,
 zugferd .initial:n = true,
 zugferd .default:n = true,
 zugferd .usage:n = load,
 ZUGFerD .meta:n = {zugferd = #1},
 xml-file .tl_gset:N = \g__zugferd_xml_file_tl,
 xml-file .initial:n = \jobname _zugferd.xml,
 xml-file .usage:n = load,
 auto-exemption .bool_gset:N = \g__zugferd_auto_exemption_bool,
 auto-exemption .initial:n = true,
 auto-exemption .default:n = true,
 auto-exemption .usage:n =load,
}
\char_set_catcode_parameter:N \#%

\ProcessKeyOptions[zugferd]
```

## 6.1 Preparation to write the `.xml` file

__zugferd_xml_writer_iow

```
\iow_new:N \__zugferd_xml_writer_iow
```

(*End of definition for* `\__zugferd_xml_writer_iow`*.*)

To adjust the metadata it is necessary to use the pdfmanagement-testphase by **pdfmanagement-testphase**. She had prepared some experiment files for the PDF attachment in the experiments of the repository. We use these to embed the XML file.

This part prepares the XMP metadata according to the required scheme.

```
\bool_if:NT \g__zugferd_active_bool {
 \cs_if_exist:NF \pdfmeta_xmp_xmlns_new:nn {
  \msg_new:nnnn {zugferd} {PDFmanagement-not-active} {
   The~\LaTeX~PDF~management~is~not~active.\\
   Activate~it~using~\string\DocumentMetadata.
  } {
   See~ZUGFeRD~or~PDFmanagement~documentation~for~more~information.
  }
  \msg_error:nn{zugferd} {PDFmanagement-not-active}
 }
 %% based on experiments for l3pdfmeta by Ulrike Fischer
 \pdfmeta_xmp_xmlns_new:nn {fx}{
  urn:factur-x:pdfa:CrossIndustryDocument:invoice:1p0\c_hash_str
 }

 \pdfmeta_xmp_schema_new:nnn
    {Factur-X~PDFA~Extension~Schema}
    {fx}
    {urn:factur-x:pdfa:CrossIndustryDocument:invoice:1p0\c_hash_str}

 \pdfmeta_xmp_property_new:nnnnn
    {fx}
    {DocumentFileName}
    {Text}
    {external}
    {name~of~the~embedded~XML~invoice~file}

 \pdfmeta_xmp_property_new:nnnnn
    {fx}
    {DocumentType}
    {Text}
    {external}
    {INVOICE}

 \pdfmeta_xmp_property_new:nnnnn
    {fx}
    {Version}
    {Text}
    {external}
    {The~actual~version~of~the~factur-x~schema}

 \pdfmeta_xmp_property_new:nnnnn
    {fx}
    {ConformanceLevel}
    {Text}
    {external}
    {The~conformance~level~of~the~factur-x~data}

 \exp_args:Ne \pdfmeta_xmp_add:n {
  % fix INVOICE
  <fx:DocumentType>INVOICE</fx:DocumentType>\iow_newline:
  % fix factur-x.xml
  <fx:DocumentFileName>factur-x.xml</fx:DocumentFileName>\iow_newline:
```

```
  % fix schema version
  <fx:Version>1.0</fx:Version>\iow_newline:
  % zulässige Werte MINIMUM, BASIC WL, BASIC, EN 16931, EXTENDED, XRECHNUNG
  <fx:ConformanceLevel>\g__zugferd_conformance_level_str</fx:ConformanceLevel>%
  \iow_newline:
  %
 }
}
```

\SetZUGFeRDData

```
\NewDocumentCommand{\SetZUGFeRDData}{sm}{
 \IfBooleanTF{#1}
  {\keys_set:ne}
  {\keys_set:nn}
   {zugferd} {#2}
}
\let\SetZugferdData\SetZUGFeRDData
```

\InsertZUGFeRDData
\InsertZugferdData

To simplify the usage in letters we also add fields to be able to use the zugferd data within LaTeX output. Country is still missing.

```
\NewDocumentCommand{\InsertZUGFeRDData}{om}{
 \str_case:nnF {#1} {
  {AddressData} {
   \clist_map_inline:nn {name, lineone, linetwo} {
    \prop_if_in:cnT {g__zugferd_#2_AddressData_prop } {##1} {
     \prop_item:cn {g__zugferd_#2_AddressData_prop } {##1}\\
    }
   }
   \prop_item:cn {g__zugferd_#2_AddressData_prop } {postcode}
   \space
   \prop_item:cn {g__zugferd_#2_AddressData_prop } {city}
  }
  {set-today} {
   \__zugferd_set_today:v {g__zugferd_#2_tl}
  }
 } {
```

Try to find the variable automatially.

- replace dashes by underscores
- try if a tokenlist or a string
- prefer global over local
- take the first existing variable and break the loop

```
  \str_set:Ne \l_tmpa_str {#2}
  \str_replace_all:Nnn \l_tmpa_str {-} {_}
  \str_replace_all:Nnn \l_tmpa_str {/} {_}
  \bool_set_true:N \g_tmpa_bool
  \clist_map_inline:nn {tl, str} {
   \clist_map_inline:nn {g, l} {
    \use:c {##1 _if_exist:cT} {####1__zugferd_ \l_tmpa_str _##1}
     {
      \use:c {####1__zugferd_ \l_tmpa_str _##1}
      \bool_gset_false:N \g_tmpa_bool
     }
    \bool_if:NF \g_tmpa_bool {\clist_map_break:}
   }
   \bool_if:NF \g_tmpa_bool {\clist_map_break:}
  }
 }
}
\providecommand{\InsertZugferdData}{\InsertZUGFeRDData}
\providecommand{\insertZugferdData}{\InsertZUGFeRDData}
```

`\__zugferd_set_today:c` Auxiliary function to use a date variable within the current group to be used as `\today`.

```
\cs_new:Nn \__zugferd_set_today:n {
 \__zugferd_set_today_aux:w #1 \q_stop
}
\cs_generate_variant:Nn \__zugferd_set_today:n {v}
\cs_new:Npn \__zugferd_set_today_aux:w #1 #2 #3 #4 #5 #6 #7 #8 \q_stop{
 \int_set:Nn \year {#1#2#3#4}
 \int_set:Nn \month {#5#6}
 \int_set:Nn \day {#7#8}
}
```

```
\NewDocumentEnvironment{ZUGFeRD}{o}{
 \IfNoValueF{#1}{
  \SetZUGFeRDData{#1}
 }
 \zugferd_enable_XML_interfaces:
 \startWritingZUGFeRDxml
 \zugferd_write_Header:
 \ignorespaces
}{
 \zugferd_enable_XML_interfaces:
 \zugferd_write_Footer:
 \stopWritingZUGFeRDxml
 \zugferd_disable_XML_interfaces:
}
```

```
\newcommand*{\startWritingZUGFeRDxml}{
 \begingroup
 \bool_if:NTF \g__zugferd_write_xml_bool {
  \char_set_active_eq:nN {13} \__zugferd_xml_newline_indent:
  \iow_open:Nn \__zugferd_xml_writer_iow {\g__zugferd_xml_file_tl}
 }{
  \msg_info:nn {zugferd} {no-xml-write}
 }
}
\msg_new:nnn {zugferd} {no-xml-write} {
 The~option~write-xml=false~was~set.\\
 Writing~of~XML~file~is~deactivated.
}
```

The PDF attachment is done after the writing stream is closed.

```
%% The metadata elements are taken by Ulrike Fischer's faktur-x experiments
%% https://github.com/latex3/pdfresources/tree/main/experiments/factur-x-bills
\newcommand*{\stopWritingZUGFeRDxml}{%
 \bool_if:NT \g__zugferd_write_xml_bool
  {\iow_close:N \__zugferd_xml_writer_iow}%
 \endgroup
 \bool_if:NT \g__zugferd_active_bool {
  \group_begin:
   \pdfdict_put:nnn {l_pdffile/Filespec} {AFRelationship}{/Alternative}
   %or /Source in some cases
   \pdfdict_put:nnn {l_pdffile/Filespec} {Desc}{(Factur-X/ZUGFeRD-Rechnung)}
   \pdffile_embed_file:nnn {\g__zugferd_xml_file_tl}{factur-x.xml}
    {zugferd/rechnung}
  \group_end:
  \pdfmanagement_add:nnx
   {Catalog/Names}
   {EmbeddedFiles}
   {\pdf_object_ref:n{zugferd/rechnung}}
   % steht in der docu ist aber pdf 2.0 ....
  \pdfmanagement_add:nnx{Catalog}{AF}{\pdf_object_ref:n{zugferd/rechnung}}
 }
}
```

Provide public interfaces and the ZUGFeRD environment.

\zugferd_enable_XML_interfaces:
\zugferd_disable_XML_interfaces:

```
\cs_new:Nn \zugferd_enable_XML_interfaces: {
 \bool_if:NT \g__zugferd_write_xml_bool {
  \cs_gset:Nn \zugferd_write_Header: {
   \__zugferd_insert_Header:
   \__zugferd_insert_FrontMatter:
  }
  \cs_gset:Nn \zugferd_write_Footer: {
   \__zugferd_insert_Footer:
  }
  \cs_gset_eq:NN \zugferd_write_Item:nnnnnn
   \__zugferd_insert_TradeLineItem:nnnnnn
  \cs_gset:Nn \zugferd_startInvoiceSums: {
   \__zugferd_ApplicableHeaderTradeAgreement:
   \__zugferd_ApplicableHeaderTradeSettlement_start:
   \__zugferd_SpecifiedTradeSettlementPaymentMeans:
  }
  \cs_gset:Nn \zugferd_stopInvoiceSums:  {
   \__zugferd_ApplicableHeaderTradeSettlement_stop:
  }
  \cs_gset_eq:NN \zugferd_write_TaxEntry:nnnn \__zugferd_ApplicableTradeTax:nnnn
  \cs_gset:Nn \zugferd_write_Summation:nnnnnnnn {
   \__zugferd_SpecifiedTradePaymentTerms:
   \__zugferd_SpecifiedTradeSettlementHeaderMonetarySummation:nnnnnnnn
    {##1} {##2} {##3} {##4} {##5} {##6} {##7} {##8}
  }
 }
}
\cs_new:Nn \zugferd_disable_XML_interfaces: {
 \cs_gset_eq:NN \zugferd_write_Header: \prg_do_nothing:
 \cs_gset_eq:NN \zugferd_write_Footer: \prg_do_nothing:
 \cs_gset_eq:NN \zugferd_write_Item:nnnnnn \use_none:nnnnnn
 \cs_gset_eq:NN \zugferd_startInvoiceSums: \prg_do_nothing:
 \cs_gset_eq:NN \zugferd_stopInvoiceSums: \prg_do_nothing:
 \cs_gset_eq:NN \zugferd_write_TaxEntry:nnnn \use_none:nnnn
 \cs_gset_eq:NN \zugferd_write_Summation:nnnnnnnn \use_none:nnnnnnnn
}
\bool_if:NTF \g__zugferd_write_xml_bool {
 \zugferd_enable_XML_interfaces:
}{
 \zugferd_disable_XML_interfaces:
}
```

__zugferd_write_Item:nnnnnnn

```
\cs_new:Nn \zugferd_write_Item:nnnnnnn {
 \group_begin:
 \keys_set:nn {zugferd/item} {#1}
 \zugferd_write_Item:nnnnnn {#2} {#3} {#4} {#5} {#6} {#7}
 \group_end:
}
\cs_generate_variant:Nn \zugferd_write_Item:nnnnnnn {ennnnnn}
```

```
\__zugferd_write_xml:n
\__zugferd_define_xml_writer:Nn
\__zugferd_define_xml_content:Nn
```

These commands are used to toggle the writing of the XML file. This corresponds to the option `write-xml`.

```
\bool_if:NTF \g__zugferd_write_xml_bool {
 \cs_new:Nn \__zugferd_write_xml:n {
  \iow_now:Ne \__zugferd_xml_writer_iow {\__zugferd_xml_auto_indent: #1}
 }
 \cs_new_eq:NN \__zugferd_define_xml_writer:Nn \cs_new:Nn
 \cs_new_eq:NN \__zugferd_define_xml_content:Nn \cs_new:Nn
} {
 \cs_set_eq:NN \__zugferd_write_xml:n \use_none:n
 \cs_set:Nn \__zugferd_define_xml_writer:Nn {\cs_new:Nn #1 {}}
 \cs_set:Nn \__zugferd_define_xml_content:Nn {\cs_new:Nn #1 {}}
}

\cs_generate_variant:Nn \__zugferd_write_xml:n {e}
```

## 6.2 Number rounding

As siunitx is implementing this, we use it instead of building our own mechanism.

```
\RequirePackage{siunitx}
```

```
\__zugferd_number_format:nNn
\__zugferd_number_format:nNe
```

```
\cs_new:Nn \__zugferd_number_format:nNn {
 \sisetup{
  parse-numbers=true,
  round-mode=places,
  round-precision=#1,
  round-pad = false,
  group-digits=false,
  minimum-decimal-digits=#1,
  output-decimal-marker=.
 }
 \siunitx_number_format:nN {#3} #2
}
\cs_generate_variant:Nn \__zugferd_number_format:nNn {nNe}
```

```
\__zugferd_write_rounded:nnnn
\__zugferd_write_rounded:nnn
```

```
\cs_new:Nn \__zugferd_write_rounded:nnnn {
 \__zugferd_number_format:nNe {#1} \l__zugferd_tmp_tl {#4}
 \__zugferd_write_xml:e {<ram:#2#3>\l__zugferd_tmp_tl</ram:#2>}
}
\cs_new:Nn \__zugferd_write_rounded:nnn {
 \__zugferd_write_rounded:nnnn {#1} {#2} {} {#3}
}
```

## 6.3 XML indentation

The indentation of the XML does not really matter. For debugging, it's a lot simpler to have it included and this also helped to maintain the structure of the code during development, so I decided to keep it. The indentation is created using s bunch of auxiliary commands and variables which are defined here.

\g__zugferd_indent_int

```
\int_new:N \g__zugferd_indent_int
```

(*End of definition for \g__zugferd_indent_int.*)

__zugferd_indent:
__zugferd_xml_auto_indent:
__zugferd_xml_newline_indent:

```
\cs_new:Nn \__zugferd_indent: {
 \space\space
}
\cs_new:Nn \__zugferd_indent:n {
 \prg_replicate:nn {#1} {\__zugferd_indent:}
}
\cs_new:Nn \__zugferd_xml_auto_indent: {
 \__zugferd_indent:n {\g__zugferd_indent_int}
}
```

The idea was to redefine the \newlinechar to automatically indent the following line.

```
\cs_new:Nn \__zugferd_xml_newline_indent: {
 \iow_newline: \__zugferd_xml_auto_indent:
}
```

Setting up the catcodes to simplify the XML indentation.

\zugferd_disable_macros:

The definition was mostly taken from hyperref [**hyperref**]. Most likely not all of these are required, but it's probably easier to take this as a reasonable choice instead of creating an own collection.

```
\cs_new:Nn \zugferd_disable_macros: {
 \let\{\textbraceleft
 \let\}\textbraceright
 \let\\\textbackslash
 \let\#\textnumbersign
 \let\$\textdollar
```

This only is a part of the list. There is no real use of printing the whole list, it's inside `zugferd.sty` anyway.

\g__zugferd_notes_seq
\g__zugferd_id_tl
\g__zugferd_date_tl
\g__zugferd_subject_tl
\g__zugferd_fromaddress_tl
\g__zugferd_DocumentTypeCode_tl

```
\seq_new:N \g__zugferd_notes_seq
\tl_new:N \g__zugferd_id_tl
\tl_new:N \g__zugferd_date_tl
\tl_new:N \g__zugferd_delivery_date_tl
\tl_new:N \g__zugferd_subject_tl
\tl_new:N \g__zugferd_fromaddress_tl
```

```
\tl_new:N \g__zugferd_DocumentTypeCode_tl
\tl_new:N \l__zugferd_currency_tl
```

(*End of definition for \g__zugferd_notes_seq and others.*)

```
\keys_define:nn { zugferd } {
%TODO define others
```

id
document-type

```
 document-type .choice:,
 document-type / commercial-invoice .code:n =
  \tl_gset:Nn \g__zugferd_DocumentTypeCode_tl {380},
 document-type .initial:n = commercial-invoice,
 id .choice:,
 id / komavar .code:n = \tl_gset:Nf \g__zugferd_id_tl {\scr@invoice@var},
 id / unknown .code:n = \tl_gset:Nn \g__zugferd_id_tl {#1},
 currency .choices:nn = {EUR, USD, CHF} {
  \tl_set_eq:NN \l__zugferd_currency_tl \l_keys_choice_tl
 },
 currency / € .meta:n = {currency = EUR},
 currency / unknown .code:n = {
  \exp_args:Nnnx \keys_if_choice_exist:nnnTF {zugferd} {unit} {
   \str_uppercase:f {#1}
  }
  {\keys_set:nx { zugferd } {unit= {\str_uppercase:f {#1}}}}
  {
   \msg_warning:nnn { zugferd } {unknown-value} {currency} {#1}
   \tl_set:Ne \l__zugferd_currency_tl {\str_uppercase:f {#1}}
  }
 },
 currency .initial:n = EUR,
```

date
delivery-date

```
 date .choice:,
 date / auto .code:n = \tl_gset:Ne \g__zugferd_date_tl {
  \the\year
  \int_compare:nNnT {\month} < {10} {0} \the\month
  \int_compare:nNnT {\day} < {10} {0}\the\day
 },
 date / unknown .code:n = \tl_gset:Nn \g__zugferd_date_tl {#1},
 date .initial:n = auto,
 delivery-date .choice:,
 delivery-date / auto .code:n = \tl_gset:Ne \g__zugferd_delivery_date_tl {
  \the\year
  \int_compare:nNnT {\month} < {10} {0} \the\month
  \int_compare:nNnT {\day} < {10} {0}\the\day
 },
 delivery-date / unknown .code:n = {
  \tl_gset:Nn \g__zugferd_delivery_date_tl {#1}
 },
 delivery-date .initial:n = auto,
 due-date .tl_gset:N = \g__zugferd_due_date_tl,
 due-date .initial:n =,
```

payment-terms

```
 payment-terms .str_gset:N = \g__zugferd_payment_terms_str,
 payment-terms .initial:n =,
```

subject
fromaddress
add-note

```
 subject .choice:,
 subject / komavar .code:n = {
  \tl_gset:Nf \g__zugferd_subject_tl {\scr@subject@var}
 },
 subject / unknown .code:n  = \tl_gset:Nn \g__zugferd_subject_tl {#1},
 fromaddress .choice:,
 fromaddress / komavar .code:n = \tl_gset:Nf \g__zugferd_fromaddress_tl
  {\scr@fromaddress@var},
 fromaddress / unknown .code:n = \tl_gset:Nn \g__zugferd_fromaddress_tl {#1},
 add-note .code:n = \seq_gput_right:Nn \g__zugferd_notes_seq {#1},
}
\msg_new:nnnn {zugferd} {unknown-value} {
 You~selected~a~#1~which~was~not~predefined.\\
 I~will~directly~use~your~selection~'#1=#2'.
 Please~ensure~the~selection~is~valid!
} {
 For~more~information~see~the~zugferd~documentation.
}
```

Macro to write notes

```
\cs_new:Nn \__zugferd_note_if_not_empty:N {%
 \tl_if_empty:NF #1 {%
  \__zugferd_write_note:n {#1}%
 }%
}%
```

ApplicableHeaderTradeAgreement

Contains information on seller and buyer trade party:

- BuyerReference
- SellerTradeParty
- BuyerTradeParty

\metaparty/address
\g__zugferd_seller_AddressData_prop
\g__zugferd_buyer_AddressData_prop
\g__zugferd_shipto_AddressData_prop

Seller and Buyer are specified the same way.

```
\clist_map_inline:nn {seller,buyer,shipto} {
 \prop_new:c {g__zugferd_#1_AddressData_prop}
 \keys_define:nn {zugferd / #1} {
  address .code:n = {
   \seq_set_split:Nnn \l_tmpa_seq {\\} {##1}
   %TODO error if more than 2 lines or only 1
   \keys_set:nx {zugferd / #1} {
    lineone=\seq_item:Nn \l_tmpa_seq {1},
    linetwo=\seq_item:Nn \l_tmpa_seq {2}
   }
  },
  contact .code:n = {
   \seq_set_split:Nnn \l_tmpa_seq {\\} {##1}
   \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > {3} {
    \keys_set:ne {zugferd/#1} {
     contact-name = \seq_item:Nn \l_tmpa_seq {1},
     contact-department = \seq_item:Nn \l_tmpa_seq {2},
     contact-phone = \seq_item:Nn \l_tmpa_seq {3},
     contact-email= \seq_item:Nn \l_tmpa_seq {4}
    }
   }{
    \keys_set:ne {zugferd/#1} {
     contact-name = \seq_item:Nn \l_tmpa_seq {1},
     contact-phone = \seq_item:Nn \l_tmpa_seq {2},
     contact-email= \seq_item:Nn \l_tmpa_seq {3}
    }
   }
  },
  unknown .code:n = \tl_if_blank:nF {##1} {
   \prop_gput:cVe {g__zugferd_#1_AddressData_prop}
    \l_keys_key_tl {\tl_trim_spaces:n {##1}}
  }
 }
}
```

(*End of definition for \g__zugferd_seller_AddressData_prop , \g__zugferd_buyer_AddressData_prop , and \g__zugferd_shipto_AddressData_prop.*)

__zugferd_PostalTradeAddress:N
__zugferd_DefinedTradeContact:N

Wrappers to map the property list items to the writing macro.

```
\cs_new:Nn \__zugferd_PostalTradeAddress_short:N {
 \exp_args:Ne \tl_if_blank:nF {\prop_item:Nn #1 {name}}
 {<ram:Name>\prop_item:Nn #1 {name}</ram:Name>\iow_newline:\__zugferd_xml_auto_indent:}
 \__zugferd_PostalTradeAddress:eeeee
  {\prop_item:Nn #1 {postcode}}
  {\prop_item:Nn #1 {lineone}}
  {\prop_item:Nn #1 {linetwo}}
  {\prop_item:Nn #1 {city}}
  {\prop_item:Nn #1 {country}}
}
\cs_new:Nn \__zugferd_PostalTradeAddress:N {%
 \__zugferd_PostalTradeAddress:eeeeeee
  {\prop_item:Nn #1 {postcode}}
  {\prop_item:Nn #1 {lineone}}
  {\prop_item:Nn #1 {linetwo}}
  {\prop_item:Nn #1 {city}}
  {\prop_item:Nn #1 {country}}
  {\prop_item:Nn #1 {email}}
  {\prop_item:Nn #1 {vatid}}
}%
\cs_new:Nn \__zugferd_DefinedTradeContact:N {%
 \__zugferd_DefinedTradeContact:eeee
  {\prop_item:Nn #1 {contact-name}}
  {\prop_item:Nn #1 {contact-department}}
  {\prop_item:Nn #1 {contact-phone}}
  {\prop_item:Nn #1 {contact-email}}
}%
```

```
\cctab_begin:N \g__zugferd_xml_cctab%
\__zugferd_define_xml_content:Nn \__zugferd_PostalTradeAddress:nnnnn {%
 <ram:PostalTradeAddress>
 \__zugferd_indent: <ram:PostcodeCode>#1</ram:PostcodeCode>
 \tl_if_blank:nF {#2} {%
  \__zugferd_indent: <ram:LineOne>#2</ram:LineOne>
 }%
 \tl_if_blank:nF {#3} {%
  \__zugferd_indent: <ram:LineTwo>#3</ram:LineTwo>
 }%
 \__zugferd_indent: <ram:CityName>#4</ram:CityName>
 \__zugferd_indent: <ram:CountryID>#5</ram:CountryID>
 </ram:PostalTradeAddress>%
}
\__zugferd_define_xml_content:Nn \__zugferd_PostalTradeAddress:nnnnnnn {%
 \__zugferd_PostalTradeAddress:nnnnn {#1} {#2} {#3} {#4} {#5}
 \tl_if_empty:nF {#6} {%
  <ram:URIUniversalCommunication>
  \__zugferd_indent: <ram:URIID~schemeID="EM">#6</ram:URIID>
  </ram:URIUniversalCommunication>%
 }%
 % TODO add support local tax id: schemaID="FC"
 \tl_if_empty:nF {#7} {
  <ram:SpecifiedTaxRegistration>
  \__zugferd_indent: <ram:ID~schemeID="VA">#7</ram:ID>
  </ram:SpecifiedTaxRegistration>%
 }%
}%
%
% Contact data phone/email to a specific contact person
\__zugferd_define_xml_content:Nn \__zugferd_DefinedTradeContact:nnnn {%
 % Do not print if name is empty
 \tl_if_blank:nT {#1} {\use_none:nnn} %
 \bool_if:NT \g__zugferd_writeTradeContact_bool {%
  <ram:DefinedTradeContact>
  \__zugferd_indent: <ram:PersonName>#1</ram:PersonName>
  \tl_if_blank:nF {#2} {%
   \__zugferd_indent: <ram:DepartmentName>#2</ram:DepartmentName>
  }%
  \tl_if_blank:nF {#3} {%
   \__zugferd_indent: <ram:TelephoneUniversalCommunication>
   \__zugferd_indent:n {2} <ram:CompleteNumber>#3</ram:CompleteNumber>
   \__zugferd_indent: </ram:TelephoneUniversalCommunication>
  }%
  \tl_if_blank:nF {#4} {%
   \__zugferd_indent: <ram:EmailURIUniversalCommunication>
   \__zugferd_indent:n {2} <ram:URIID>#4</ram:URIID>
   \__zugferd_indent: </ram:EmailURIUniversalCommunication>
  }%
  </ram:DefinedTradeContact>%
 }%
}%
%
\__zugferd_define_xml_writer:Nn \__zugferd_ApplicableHeaderTradeAgreement: {%
```

```
 \__zugferd_write_xml:n {<ram:ApplicableHeaderTradeAgreement>}%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  <ram:BuyerReference>\g__zugferd_buyer_reference_tl</ram:BuyerReference>
  <ram:SellerTradeParty>%
 }%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  <ram:Name>\prop_item:Nn \g__zugferd_seller_AddressData_prop {name}</ram:Name>
  \__zugferd_DefinedTradeContact:N \g__zugferd_seller_AddressData_prop%
  \__zugferd_PostalTradeAddress:N \g__zugferd_seller_AddressData_prop%
 }%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  </ram:SellerTradeParty>
  <ram:BuyerTradeParty>%
 }%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  <ram:Name>\prop_item:Nn \g__zugferd_buyer_AddressData_prop {name}</ram:Name>
  \__zugferd_DefinedTradeContact:N \g__zugferd_buyer_AddressData_prop%
  \__zugferd_PostalTradeAddress:N \g__zugferd_buyer_AddressData_prop%
 }%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:BuyerTradeParty>}%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:ApplicableHeaderTradeAgreement>}%
 \__zugferd_ApplicableHeaderTradeDelivery:V \g__zugferd_delivery_date_tl%
}%
%
%
% delivery date
\__zugferd_define_xml_content:Nn \__zugferd_ApplicableHeaderTradeDelivery:n {%
 \bool_lazy_and:nnF {\prop_if_empty_p:N \g__zugferd_shipto_AddressData_prop}%
   {\tl_if_blank_p:n {#1}}%
 {%
  \__zugferd_write_xml:n {<ram:ApplicableHeaderTradeDelivery>}%
  \prop_if_empty:NF \g__zugferd_shipto_AddressData_prop {%
   \int_gincr:N \g__zugferd_indent_int%
   \__zugferd_write_xml:n {<ram:ShipToTradeParty>}%
   \int_gincr:N \g__zugferd_indent_int%
   \__zugferd_write_xml:e {%
    \__zugferd_PostalTradeAddress_short:N \g__zugferd_shipto_AddressData_prop%
   }%
   \int_gdecr:N \g__zugferd_indent_int%
   \__zugferd_write_xml:n {</ram:ShipToTradeParty>}%
   \int_gdecr:N \g__zugferd_indent_int%
  }%
  \tl_if_blank:nF {#1} {%
   \__zugferd_write_xml:n {%
    \__zugferd_indent: <ram:ActualDeliverySupplyChainEvent>
    \__zugferd_indent:n {2} <ram:OccurrenceDateTime>
    \__zugferd_indent:n {3}<udt:DateTimeString~format="102">#1</udt:DateTimeString>
    \__zugferd_indent:n {2} </ram:OccurrenceDateTime>
    \__zugferd_indent: </ram:ActualDeliverySupplyChainEvent>%
   }%
  }%
  \__zugferd_write_xml:n {</ram:ApplicableHeaderTradeDelivery>}%
 }%
}%
%
\cctab_end:
\cs_generate_variant:Nn \__zugferd_DefinedTradeContact:nnnn {eeee}
\cs_generate_variant:Nn \__zugferd_PostalTradeAddress:nnnnn {eeeee}%
\cs_generate_variant:Nn \__zugferd_PostalTradeAddress:nnnnnnn {eeeeeee}
\cs_generate_variant:Nn \__zugferd_ApplicableHeaderTradeDelivery:n {V}
```

buyer/reference

```
\tl_new:N \g__zugferd_buyer_reference_tl
\keys_define:nn {zugferd/buyer} {
 reference .choice:,
 reference / komavar .code:n = {
  \tl_gset:Nf \g__zugferd_buyer_reference_tl {\scr@yourref@var}
 },
 reference / unknown .code:n = {
  \tl_gset:Nn \g__zugferd_buyer_reference_tl {#1}
 }
}
```

payment-means

```
\ExplSyntaxOff
\providecommand*{\zugferd@paymentMeans@german}{
  1 = Keine Zahlungsart definiert,
 10 = Bargeld,
 30 = Überweisung,
 42 = Zahlung an Bankkonto,
 48 = Kartenzahlung,
 49 = Lastschriftverfahren,
 57 = Dauerauftrag,
 58 = Zahlung per SEPA Überweisung.,
 59 = SEPA Lastschrift,
 97 = Ausgleich zwischen Partnern
}
\def\zugferd@paymentMeans@ngerman{\zugferd@paymentMeans@german}
\providecommand*{\zugferd@paymentMeans@english}{
  1 = Instrument not defined,
 10 = In cash,
 30 = Credit Transfer,
 31 = Debit Transfer,
 42 = Payment to bank account,
 48 = Bank card,
 49 = Direct Debit,
 57 = Standing agreement,
 58 = SEPA credit transfer,
 59 = SEPA direct debit,
 97 = Clearing between partners
}
\ExplSyntaxOn
```

```
\tl_new:N \g__zugferd_payment_means_tl
\keys_define:nn {zugferd} {
 payment-means / type .choices:nn = {1,10,30,42,48,49,57,58,59,97} {
  \tl_gset_eq:NN \g__zugferd_payment_means_tl \l_keys_choice_tl
 },
 payment-means / unknown .code:n = {
  \msg_warning:nnn { zugferd } {unknown-value} {payment-means} {#1}
  \tl_gset:Nn \g__zugferd_payment_means_tl {\int_eval:n {#1}}
 }
}

\clist_map_inline:nn {iban,account-holder,bic} {
 \keys_define:nn {zugferd/payment-means} {
  #1 .tl_gset:c = {g__zugferd_payment_#1_tl}
 }
}

\prop_new:c {g__zugferd_payment-means_names_prop}
\newcommand*{\setupZUGFeRDStrings}[2]{
 \prop_gset_from_keyval:cn {g__zugferd_#1_names_prop} {
  #2
 }
}

\hook_gput_code:nnn {begindocument/end}{zugferd/payment-means}{
 \prop_if_empty:cT {g__zugferd_payment-means_names_prop} {
  \exp_args:Nne \setupZUGFeRDStrings{payment-means}{
   \use:c {zugferd@paymentMeans@
    \cs_if_exist:cTF {zugferd@paymentMeans@\languagename} {\languagename} {english}
   }
  }
 }
}
```

unit

```
\keys_define:nn { zugferd } {
 unit .choices:nn = {HUR,DAY,C62,H87} {
  \tl_set_eq:NN \l__zugferd_unit_code_tl \l_keys_choice_tl
 },
 unit / hour .meta:n = {unit=HUR},
 unit / day .meta:n = {unit=DAY},
 unit / one .meta:n= {unit= C62},
 unit / piece .meta:n = {unit=H87},
```

If unknown, the value is converted to uppercase and we use the selection directly. There is a warning in that case

```
 unit / unknown .code:n = {
  \tl_set:Nx \l_tmpa_tl {\str_uppercase:f {#1}}
  \exp_args:NnnV \keys_if_choice_exist:nnnTF {zugferd} {unit}
   \l_tmpa_tl
  {
   \keys_set:ne { zugferd } {unit= {\l_tmpa_tl}}
  } {
   \msg_warning:nnn { zugferd } {unknown-unit} {#1}
   \tl_set_eq:NN \l__zugferd_unit_code_tl \l_tmpa_tl
  }
 },
 unit .usage:n = general,
}

\msg_new:nnnn {zugferd} {unknown-unit} {
 You~selected~a~unit~which~was~not~predefined.\\
 I~~will~directly~use~your~selection~'unit=#1'~as~Common~Code.
 Please~ensure~the~selection~is~valid!
} {
 For~more~information~see~the~zugferd~documentation\\
 and~the~/UN/CEFACT~Common~Code~list.
}
```

category
exemption-reason
rate

```
\msg_new:nnnn {zugferd} {unknown-tax-category} {
 You~selected~an~unknown~tax~category.\\
 I~~will~directly~use~your~selection~'category=#1'~as~Code.\\
 Please~ensure~the~selection~is~valid!
} {
 For~more~information~see~the~zugferd~documentation\\
 and~the~corresponding~code~list.
}
\msg_new:nnnn {zugferd} {no-auto-exemption} {
 You~selected~tax/category=#1~together~with~the~auto-exemption~option.\\
 I~don't~have~any~pre-configured~exemption~setting~for~category=#1.\\
 Please~make~sure~you~add~a~valid~setting~yourself.
} {
 For~more~information~see~the~zugferd~documentation.
}
\keys_define:nn { zugferd / tax } {
 category .choices:nn = {S,Z,E,AE,K,G,O,L,M} {
  \tl_set_eq:NN \l__zugferd_tax_category_code_tl \l_keys_choice_tl
  \bool_if:NT \g__zugferd_auto_exemption_bool {
   \keys_if_choice_exist:nnnTF {zugferd/tax}  {exemption-reason-auto} {#1} {
    \keys_set:nn  {zugferd/tax}  {exemption-reason-auto=#1}
   } {
    \msg_warning:nnn { zugferd } {no-auto-exemption} {#1}
   }
  }
 },
```

```
 exemption-reason .tl_set:N = \l__zugferd_tax_exemption_reason_tl,
 exemption-reason  .initial:V = \c_empty_tl,
 exemption-reason .usage:n = general,
 exemption-reason-code .tl_set:N  = \l__zugferd_tax_exemption_code_tl,
 exemption-reason-code  .initial:V = \c_empty_tl,
 exemption-reason .usage:n = general,
```

```
 exemption-reason-auto .choice:,
 exemption-reason-auto / S .code:n = {
  \keys_set:nn {zugferd/tax} {exemption-reason=,exemption-reason-code=}
 },
 exemption-reason-auto / K .code:n = {
  \keys_set:nn {zugferd/tax} {
   exemption-reason= Intra-Community~Supply,
   exemption-reason-code={vatex-eu-ic}
  }
 },
 exemption-reason-auto / AE .code:n = {
  \keys_set:nn {zugferd/tax}{
   exemption-reason=Reverse~Charge,
   exemption-reason-code={vatex-eu-ae}
  }
 },
 exemption-reason-auto / G .code:n = {
  \keys_set:nn {zugferd/tax}{
   exemption-reason=Export~outside~the~EU,
   exemption-reason-code={vatex-eu-g}
  }
 },
 exemption-reason-auto / O .code:n = {
  \keys_set:nn {zugferd/tax}{
   exemption-reason=No~subject~to~VAT,
   exemption-reason-code={vatex-eu-o}
  }
 },
 standard .meta:n = {category=S},
 zero .meta:n = {category=Z},
 exempt .meta:n = {category=E},
 reverse-charge .meta:n = {category=AE},
 intra-community .meta:n = {category=K},
 EEA .meta:n = {category=K},
 export .meta:n = {category=G},
 canary-islands .meta:n = {category=L},
 ceuta .meta:n = {category=M},
 melilla .meta:n = {category=M},
 category / unknown .code:n = {
  \exp_args:Nnnx \keys_if_choice_exist:nnnTF {zugferd} {category}
   {\str_uppercase:f {#1}}
  {
   \keys_set:nx { zugferd } {category= {\str_uppercase:f {#1}}}
  } {
   \msg_warning:nnn { zugferd } {unknown-tax-category} {#1}
  }
 },
 category .initial:n =  S,
 category .usage:n = general,
 rate .fp_set:N =  \l__zugferd_tax_rate_fp,
 rate .initial:n = 19,
 rate .usage:n = general
}%
\keys_define:nn {zugferd/item} {
 tax .choice:,
 tax / unknown .code:n = \keys_set:ne {zugferd/tax} { \l_keys_key_str = \exp_not:V \l_keys_v
 unknown .code:n = \keys_set:ne {zugferd} { \l_keys_key_str = \exp_not:V \l_keys_value_tl}
}
```

start-date
end-date

```
\keys_define:nn { zugferd / item } {
 start-date .tl_gset:N = \l__zugferd_start_date_tl,
 start-date .initial:n =,
 end-date .tl_gset:N = \l__zugferd_end_date_tl,
 end-date .initial:n =,
}%
\__zugferd_define_xml_writer:Nn \__zugferd_write_inline:nn {
 \tl_if_blank:nF {#2} {
  <#1>#2</#1>
 }
}
\cs_generate_variant:Nn \__zugferd_write_inline:nn {ne}
\__zugferd_define_xml_writer:Nn  \__zugferd_write_inline_i:nn {
 \__zugferd_indent: \__zugferd_write_inline:nn {#1} {#2}
}
\cctab_begin:N \g__zugferd_xml_cctab%
%
\__zugferd_define_xml_writer:Nn \__zugferd_write_note:n {%
 \begingroup%
  \let\\\iow_newline:%
  \__zugferd_write_xml:e {%
   <ram:IncludedNote>
    \__zugferd_indent: <ram:Content>
    #1
    \__zugferd_indent: </ram:Content>
   </ram:IncludedNote>%
  }%
 \endgroup%
}%
%
%
\begingroup%
\char_set_catcode_other:N \#%
\char_set_catcode:nn {32}{10}%
\__zugferd_define_xml_writer:Nn \__zugferd_insert_Header: {%
 \__zugferd_write_xml:e {%
  <?xml version='1.0' encoding='UTF-8' ?>
  <rsm:CrossIndustryInvoice %
  xmlns:rsm="urn:un:unece:uncefact:data:standard:CrossIndustryInvoice:100" %
  xmlns:qdt="urn:un:unece:uncefact:data:standard:QualifiedDataType:100" %
  xmlns:ram="urn:un:unece:uncefact:data:standard:ReusableAggregateBusinessInformationEntity:
  xmlns:xs="http://www.w3.org/2001/XMLSchema" %
  xmlns:udt="urn:un:unece:uncefact:data:standard:UnqualifiedDataType:100">%
 }%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {<rsm:ExchangedDocumentContext>}%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  \str_if_empty:NF \g__zugferd_businessProcessId_str {%
   <ram:BusinessProcessSpecifiedDocumentContextParameter>
   \__zugferd_indent: <ram:ID>\g__zugferd_businessProcessId_str</ram:ID>
   </ram:BusinessProcessSpecifiedDocumentContextParameter>
  }%
  <ram:GuidelineSpecifiedDocumentContextParameter>
  \__zugferd_indent: <ram:ID>\g__zugferd_format_str</ram:ID>
  </ram:GuidelineSpecifiedDocumentContextParameter>%
 }%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</rsm:ExchangedDocumentContext>}%
}%
\endgroup%
\__zugferd_define_xml_writer:Nn \__zugferd_insert_FrontMatter: {%
 \__zugferd_write_xml:n {<rsm:ExchangedDocument>}%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  <ram:ID>\g__zugferd_id_tl</ram:ID>
  <ram:TypeCode>\g__zugferd_DocumentTypeCode_tl</ram:TypeCode>
  <ram:IssueDateTime>
  % space required!
  \__zugferd_indent:<udt:DateTimeString~format="102">\g__zugferd_date_tl</udt:DateTimeString
  </ram:IssueDateTime>%
 }%
 \__zugferd_note_if_not_empty:N \g__zugferd_subject_tl%
 \__zugferd_note_if_not_empty:N \g__zugferd_fromaddress_tl%
 \seq_map_inline:Nn \g__zugferd_notes_seq {%
  \__zugferd_write_note:n {##1}%
 }%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  </rsm:ExchangedDocument>
  <rsm:SupplyChainTradeTransaction>%
 }%
 \int_gincr:N \g__zugferd_indent_int%
}%
%
% footer
\__zugferd_define_xml_writer:Nn \__zugferd_insert_Footer: {%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</rsm:SupplyChainTradeTransaction>}%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</rsm:CrossIndustryInvoice>}%
}%
\cctab_end:
```

## 6.4 Invoice Items

Each item consists of 5 parts:

- AssociatedDocumentLineDocument
- SpecifiedTradeProduct
- SpecifiedLineTradeAgreement
- SpecifiedLineTradeDelivery
- SpecifiedLineTradeSettlement

These are implemented as separate commands to be more flexible. The wrapper command is called `\__zugferd_insert_TradeLineItem:nnnnnn` and is created to be used in your own invoicing package-

```
\cctab_begin:N \g__zugferd_xml_cctab%
```

`\__zugferd_AssociatedDocumentLineDocument:n`

```
\__zugferd_define_xml_writer:Nn \__zugferd_AssociatedDocumentLineDocument:n {%
 <ram:AssociatedDocumentLineDocument>
 \__zugferd_indent: <ram:LineID>#1</ram:LineID>
 </ram:AssociatedDocumentLineDocument>%
}%
```

`\__zugferd_SpecifiedTradeProduct:nn`

```
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedTradeProduct:nn {%
 <ram:SpecifiedTradeProduct>
 \tl_if_empty:nF {#1} {%
  \__zugferd_indent: <ram:SellerAssignedID>#1</ram:SellerAssignedID>
 }%
 \__zugferd_indent: <ram:Name>#2</ram:Name>
 </ram:SpecifiedTradeProduct>%
}%
```

`\__zugferd_ProductTradePrice:nn`

```
\__zugferd_define_xml_writer:Nn \__zugferd_ProductTradePrice:nn {%
 <ram:\str_uppercase:n #1PriceProductTradePrice>
 \__zugferd_indent: <ram:ChargeAmount>#2</ram:ChargeAmount>
 </ram:\str_uppercase:n #1PriceProductTradePrice>%
}%
```

`\__zugferd_SpecifiedLineTradeAgreement:nn`

```
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedLineTradeAgreement:nn {%
 <ram:SpecifiedLineTradeAgreement>
 \__zugferd_indent:<ram:GrossPriceProductTradePrice>
 \__zugferd_indent:n {2} <ram:ChargeAmount>#1</ram:ChargeAmount>
 \__zugferd_indent:</ram:GrossPriceProductTradePrice>
 \__zugferd_indent:<ram:NetPriceProductTradePrice>
 \__zugferd_indent:n {2} <ram:ChargeAmount>#2</ram:ChargeAmount>
 \__zugferd_indent:</ram:NetPriceProductTradePrice>
 </ram:SpecifiedLineTradeAgreement>
}%
```

### 6.4.1 SpecifiedLineTradeDelivery

:nn <unit code > <number>

__zugferd_SpecifiedLineTradeDelivery:nn

```
\__zugferd_define_xml_content:Nn \__zugferd_SpecifiedLineTradeDelivery:nn {%
 <ram:SpecifiedLineTradeDelivery>
 % SPACE!
 \__zugferd_indent: <ram:BilledQuantity~unitCode="#1">#2</ram:BilledQuantity>
 </ram:SpecifiedLineTradeDelivery>
}%
```

__zugferd_SpecifiedLineTradeSettlement:nnn
__zugferd_SpecifiedLineTradeSettlement:Vnn

```
\__zugferd_define_xml_writer:Nn \__zugferd_write_SpecifiedLineTradeSettlement:nnn {%
 \__zugferd_write_xml:n {<ram:SpecifiedLineTradeSettlement>}%
 \int_gincr:N \g__zugferd_indent_int%
  \__zugferd_write_xml:e {\__zugferd_Line_ApplicableTradeTax:nn {#1} {#2}}%
  \__zugferd_write_xml:e {\__zugferd_BillingSpecifiedPeriod:VV \l__zugferd_start_date_tl \l_
  \__zugferd_write_xml:e {\__zugferd_SpecifiedTradeSettlementLineMonetarySummation:n {#3}}%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:SpecifiedLineTradeSettlement>}%
}%
\cs_generate_variant:Nn \__zugferd_write_SpecifiedLineTradeSettlement:nnn {Vnn}%
\__zugferd_define_xml_content:Nn \__zugferd_Line_ApplicableTradeTax:nn {%
 <ram:ApplicableTradeTax>
 %BT-151
 \__zugferd_indent: <ram:TypeCode>VAT</ram:TypeCode>
 %BT-151
 \__zugferd_indent:  <ram:CategoryCode>#1</ram:CategoryCode>
 %BT-152
 \__zugferd_indent: <ram:RateApplicablePercent>#2</ram:RateApplicablePercent>
 </ram:ApplicableTradeTax>
}%
\__zugferd_define_xml_content:Nn \__zugferd_BillingSpecifiedPeriod:nn {%
 \bool_lazy_or:nnF {\tl_if_blank_p:n {#1}} {\tl_if_blank_p:n {#2}} {%
 <ram:BillingSpecifiedPeriod>
 %<!-- BT-134 -->
 \__zugferd_indent:<ram:StartDateTime>
 \__zugferd_indent:n {2} <udt:DateTimeString~format="102">#1</udt:DateTimeString>
 \__zugferd_indent:</ram:StartDateTime>
 %<!-- BT-135 -->
 \__zugferd_indent:<ram:EndDateTime>
 \__zugferd_indent:n {2} <udt:DateTimeString~format="102">#2</udt:DateTimeString>
 \__zugferd_indent:</ram:EndDateTime>
     </ram:BillingSpecifiedPeriod>
     }%
}%
\__zugferd_define_xml_content:Nn \__zugferd_SpecifiedTradeSettlementLineMonetarySummation:n
 <ram:SpecifiedTradeSettlementLineMonetarySummation>
 % BT-131
 \__zugferd_indent: <ram:LineTotalAmount>#1</ram:LineTotalAmount>
  </ram:SpecifiedTradeSettlementLineMonetarySummation>
}

\cctab_end:
\cs_generate_variant:Nn \__zugferd_SpecifiedLineTradeSettlement:nnn {Vnn}
\cs_generate_variant:Nn \__zugferd_BillingSpecifiedPeriod:nn {VV}
```

The exemption reason was placed wrong with the pre-CTAN release. This was fixed in August 2024. The old macro will be kept for a bit longer but will be removed soon. Please update your own implementations accordingly.

```
\cs_new:Nn \__zugferd_SpecifiedLineTradeSettlement:nnnn {
 \msg_warning:nnnn {zugferd} {macro-deprecated}
```

```
  { \__zugferd_SpecifiedLineTradeSettlement:nnnn }
  { \__zugferd_SpecifiedLineTradeSettlement:nnn }
 \__zugferd_SpecifiedLineTradeSettlement:nnn {#2} {#3} {#4}
}
\cs_generate_variant:Nn \__zugferd_SpecifiedLineTradeSettlement:nnnn {VVnn}
```

`\__zugferd_insert_TradeLineItem:nnnnnn`
`\__zugferd_write_TradeLineItem:nnnnnn`

Writing an invoice item using the helper commands defined before, to the XML file.

```
\__zugferd_define_xml_writer:Nn \__zugferd_insert_TradeLineItem:nnnnnn {
 \__zugferd_write_xml:n {<ram:IncludedSupplyChainTradeLineItem>}
 \int_gincr:N \g__zugferd_indent_int
 \__zugferd_write_xml:e {\__zugferd_AssociatedDocumentLineDocument:n {#1}}%

 \begingroup
 \zugferd_disable_macros:
 \__zugferd_write_xml:e {\__zugferd_SpecifiedTradeProduct:nn {#2} {#3}}
 \endgroup

 \__zugferd_write_xml:n {<ram:SpecifiedLineTradeAgreement>}
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {\__zugferd_ProductTradePrice:nn {net} {#4}}
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:SpecifiedLineTradeAgreement>}
 \__zugferd_write_xml:e {%
  \__zugferd_SpecifiedLineTradeDelivery:nn { \l__zugferd_unit_code_tl } {#5}
 }%
 \__zugferd_write_SpecifiedLineTradeSettlement:Vnn \l__zugferd_tax_category_code_tl %
   { \fp_use:N \l__zugferd_tax_rate_fp} {#6}%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:IncludedSupplyChainTradeLineItem>}
}
```

ApplicableHeaderTradeSettlement

```
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedTradeSettlementHeaderMonetarySummation:n
 \__zugferd_write_xml:n {<ram:SpecifiedTradeSettlementHeaderMonetarySummation>}%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_rounded:nnn {2} {LineTotalAmount} {#1}
 \__zugferd_write_rounded:nnn {2} {ChargeTotalAmount} {#2}
 \__zugferd_write_rounded:nnn {2} {AllowanceTotalAmount} {#3}
 \__zugferd_write_rounded:nnn {2} {TaxBasisTotalAmount} {#4}
 \__zugferd_write_rounded:nnnn {2} {TaxTotalAmount} {~currencyID="\l__zugferd_currency_tl "}
 \__zugferd_write_rounded:nnn {2} {GrandTotalAmount} {#6}
 \__zugferd_write_rounded:nnn {2} {TotalPrepaidAmount} {#7}
 \__zugferd_write_rounded:nnn {2} {DuePayableAmount} {#8}
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:SpecifiedTradeSettlementHeaderMonetarySummation>}%
}%
% ApplicableTradeTax CategoryCode Rate BaseAmount Result
\__zugferd_define_xml_writer:Nn \__zugferd_ApplicableTradeTax:nnnn {%
 \__zugferd_write_xml:n {<ram:ApplicableTradeTax>}
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_rounded:nnn {2} {CalculatedAmount} {#4}%BT117
 \__zugferd_write_xml:n {<ram:TypeCode>VAT</ram:TypeCode>}%BT118
 \tl_if_blank:VF \l__zugferd_tax_exemption_reason_tl {%BT-120
  \__zugferd_write_xml:e {<ram:ExemptionReason>
   \l__zugferd_tax_exemption_reason_tl
   </ram:ExemptionReason>}
 }
 {{\__zugferd_write_rounded:nnn {2} {BasisAmount} {#3}}}%BT-116
 \__zugferd_write_xml:e {<ram:CategoryCode>#1</ram:CategoryCode>}%BT-118
 \tl_if_blank:VF \l__zugferd_tax_exemption_code_tl {%BT121
  \__zugferd_write_xml:e {<ram:ExemptionReasonCode>
   \l__zugferd_tax_exemption_code_tl
   </ram:ExemptionReasonCode>}
 }
 \__zugferd_write_xml:n {<ram:RateApplicablePercent>#2</ram:RateApplicablePercent>}%BT-119
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:ApplicableTradeTax>}
}%
%
\cctab_begin:N \g__zugferd_xml_cctab%
% ApplicableHeaderTradeSettlement
\__zugferd_define_xml_writer:Nn \__zugferd_ApplicableHeaderTradeSettlement:nnnnnnnn {%

}%
```

ApplicableHeaderTradeSettlement needs to be splitted to be used in separate parts of the tabular.

```
\__zugferd_define_xml_writer:Nn \__zugferd_ApplicableHeaderTradeSettlement_start: {%
 \__zugferd_write_xml:n {<ram:ApplicableHeaderTradeSettlement>}%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {<ram:InvoiceCurrencyCode>\l__zugferd_currency_tl</ram:InvoiceCurren
}%
\__zugferd_define_xml_writer:Nn \__zugferd_ApplicableHeaderTradeSettlement_stop: {%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:ApplicableHeaderTradeSettlement>}%
}%
```

SpecifiedTradePaymentTerms

```
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedTradePaymentTerms:nn {%
 \__zugferd_write_xml:n {<ram:SpecifiedTradePaymentTerms>}%
 \int_gincr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:e {%
  \tl_if_blank:nF {#1} {%
   <ram:Description>#1</ram:Description>%
  }%
  \tl_if_blank:nF {#2} {%
   <ram:DueDateDateTime>
   \__zugferd_indent: <udt:DateTimeString~format="102">#2</udt:DateTimeString>
   </ram:DueDateDateTime>%
  }%
 }%
 \int_gdecr:N \g__zugferd_indent_int%
 \__zugferd_write_xml:n {</ram:SpecifiedTradePaymentTerms>}%
}%
%
```

```
%
% sums
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedTradeSettlementPaymentMeans:nnn {%
 \bool_if:NT \g__zugferd_writePaymentMeans_bool {%
  \__zugferd_write_xml:n {<ram:SpecifiedTradeSettlementPaymentMeans>}%
  \int_gincr:N \g__zugferd_indent_int%
  \__zugferd_write_xml:e {%
   <ram:TypeCode>\g__zugferd_payment_means_tl</ram:TypeCode>
   \__zugferd_write_inline:ne {ram:Information} {\prop_item:cV {g__zugferd_payment-means_names_prop} \g__zugferd_payment_means_tl}%
   \tl_if_blank:nF {#1#2} {%
    <ram:PayeePartyCreditorFinancialAccount>
    \__zugferd_write_inline_i:nn {ram:IBANID} {#2}
    \__zugferd_write_inline_i:nn {ram:AccountName} {#1}
    </ram:PayeePartyCreditorFinancialAccount>
   }%
   \tl_if_blank:nF {#3} {%
    <ram:PayeeSpecifiedCreditorFinancialInstitution>
    \__zugferd_write_inline_i:nn {ram:BICID} {#3}
    </ram:PayeeSpecifiedCreditorFinancialInstitution>%
   }%
  }%
  \int_gdecr:N \g__zugferd_indent_int%
  \__zugferd_write_xml:n {</ram:SpecifiedTradeSettlementPaymentMeans>}%
 }%
}%
\cctab_end:
\cs_generate_variant:Nn \__zugferd_SpecifiedTradePaymentTerms:nn {VV}
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedTradePaymentTerms: {
 \__zugferd_SpecifiedTradePaymentTerms:VV \g__zugferd_payment_terms_str \g__zugferd_due_date
}
```

__zugferd_SpecifiedTradeSettlementPaymentMeans:

```
\cs_generate_variant:Nn  \__zugferd_SpecifiedTradeSettlementPaymentMeans:nnn {vvv}
\__zugferd_define_xml_writer:Nn \__zugferd_SpecifiedTradeSettlementPaymentMeans: {
 \tl_if_blank:VF \g__zugferd_payment_means_tl {
  \__zugferd_SpecifiedTradeSettlementPaymentMeans:vvv
   {g__zugferd_payment_account-holder_tl} {g__zugferd_payment_iban_tl} {g__zugferd_payment_b
 }
}
```

```
\msg_new:nnn {zugferd} {macro-deprecated} {
 The~function~#1~is~deprecated.\\
 It~was~replaced~by~#2.\\
 Please~adjust~your~mechanism~to~use~the~new~version.
}
```

# Change History

## References

[1] URL: https://xeinkauf.de/dokumente/ (visited on 08/20/2024).

[2] *ECE/TRADE/201. ISO COUNTRY CODE for Representation of Names of Countries.* URL: https://unece.org/trade/documents/iso-country-code-representation-names-countries (visited on 08/20/2024).

[3] United Nations Economic Commission for Europe (UNECE). *Code List Recommendations.* URL: https://unece.org/trade/uncefact/cl-recommendations (visited on 08/20/2024).

[4] Markus Kohm. *The scrletter package. Letter extension to KOMA-Script classes.* Version 3.41. July 7, 2023. URL: https://komascript.de/ (visited on 09/03/2024).

[5] Koodinierungsstelle für IT-Standards. *Spezifikation Standard XRechnung. CIUS und Extension.* June 20, 2024. URL: https://xeinkauf.de/app/uploads/2024/07/302-XRechnung-2024-06-20.pdf (visited on 08/20/2024).

[6] Koordinierungsstelle für IT Standards (KoSIT). *Xeinkauf FAQ. Leitweg ID.* URL: https://xeinkauf.de/faq/xrechnung#leitweg-id (visited on 09/04/2024).

[7] *Mustangproject.* URL: https://github.com/ZUGFeRD/mustangproject (visited on 08/20/2024).

[8] *Rec 20 – Codes for Units of Measure Used in International Trade.* Link directly to xlsx. [see 3, for all revisions]. URL: https://unece.org/sites/default/files/2023-10/rec20_Rev17e-2021.xlsx (visited on 08/20/2024).

[9] *VAT exemption reason code list.* URL: https://www.xrepository.de/details/urn:xoev-de:kosit:codeliste:vatex_1 (visited on 08/20/2024).

[10] *ZUGFeRD 2.2. Download page.* URL: https://ferd-net.de/standards/zugferd-2.2/zugferd-2.2.html (visited on 09/02/2024).

# Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.