The metastr Package

Niklas Beisert

Institut für Theoretische Physik
Eidgenössische Technische Hochschule Zürich
Wolfgang-Pauli-Strasse 27, 8093 Zürich, Switzerland
nbeisert@itp.phys.ethz.ch

2020/02/06, v1.0

Abstract

metastr is a \LaTeX\ package to store and compose strings in a structured way. This can serve several purposes such as: manage and write document metadata; use templates for formatting document data; assist in assembling and displaying document license information; facilitate basic internationalisation and localisation.

Contents

1 Introduction 2

2 Usage 2
  2.1 Defining Strings ........................................... 2
  2.2 Working with Strings .................................... 3
  2.3 String Variants ........................................... 4
  2.4 Write Document Metadata ................................. 5
  2.5 Copyright and Licenses .................................... 7
  2.6 Languages ................................................. 10
  2.7 Package Options .......................................... 11
  2.8 Extras .................................................... 12

3 Information 12
  3.1 Copyright .................................................. 12
  3.2 Files and Installation .................................... 13
  3.3 Related Packages ......................................... 13
  3.4 Feature Suggestions ...................................... 14
  3.5 Revision History ......................................... 14

A Sample 14

B Implementation 17
  B.1 Package Setup ........................................... 17
  B.2 Definitions ............................................... 18
  B.3 Basic Registers ......................................... 22
  B.4 Copyright and License .................................... 25
  B.5 Creative Commons ........................................ 28
  B.6 Contact Information ...................................... 33
1 Introduction

This package provides some basic functionality to store and compose strings. The main goal is to keep relevant information for the document in a structured way such that it can be accessed and used by conveniently using some standardised methods.

The package has the following goals, tasks and features:

- manage document metadata and write them to the PDF output file;
- set up and use templates for formatting document data, e.g. for title pages;
- assist in assembling and displaying document license information;
- facilitate basic internationalisation and localisation;
- provide preset texts and common license statements in different languages.

Using the structures provided by the packages makes particular sense if you can rely on predefined text and formatting or if you have a couple of similar documents for which you can define suitable templates.

2 Usage

To use the package \texttt{metastr}, add the command

\begin{verbatim}
\usepackage{metastr}
\end{verbatim}


to the preamble of the \LaTeX{} document.

2.1 Defining Strings

\begin{verbatim}
\metadef\metaset
\end{verbatim}

The package supplies registers for storing data. Registers need to be declared before they can be filled or used (unless the package option \texttt{checkdef=false} is set, see section 2.7). A new register \texttt{reg} is declared by the command:

\begin{verbatim}
\metadef{reg}
\end{verbatim}

The register \texttt{reg} can be filled with the value \texttt{def} by the command:

\begin{verbatim}
\metaset{reg}{def}
\end{verbatim}

The package declares a couple of registers for storing standard metadata. The basic set of registers consists of:

\begin{verbatim}
title document title
subtitle document subtitle
author document author
location location associated to the document
date document date
subject ‘subject’ of the document
keywords (a list of) keywords describing the document
titletext composition register for title
\end{verbatim}
The register titletext composes information for printing the title (such as title, subtitle, author, location and date). It can serve a similar purpose as the \TeX command \texttt{\textbackslash \texttt{maketitle}}.

Finally, there are some auxiliary registers:

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>language</td>
<td>main language of the document</td>
</tr>
<tr>
<td>url</td>
<td>URL of the document or additional info on it</td>
</tr>
<tr>
<td>urlmessage</td>
<td>message to describe the document URL</td>
</tr>
<tr>
<td>source</td>
<td>name of the source file</td>
</tr>
<tr>
<td>draft</td>
<td>indicator of draft version</td>
</tr>
</tbody>
</table>

The register \texttt{language} specifies the main language used in the document. This should be a two-letter language code (\texttt{ln}) potentially followed by two-letter country code (\texttt{ln-CN}) such as \texttt{en} or \texttt{en-GB}. The language has some impact on selecting register variants, see section 2.3 and section 2.6, it should therefore be set by the command:

\texttt{\textbackslash \textbackslash \texttt{metasetlang}\{ln-CN\}}

The starred version declares the language for PDF metadata rather than the document contents.

### 2.2 Working with Strings

\texttt{\textbackslash \texttt{metaget}} A register \texttt{reg} can be read out by the macro:

\texttt{\textbackslash \textbackslash \texttt{metaget}\{reg\}\{def\}}

Note that the (empty) argument in square brackets is mandatory, it cannot be left out, see section 2.3 for further details on its purpose. This is because \texttt{\textbackslash \texttt{metaget}} must be robust so that its output can be processed for writing (optional arguments make a macro fragile). In case the register \texttt{reg} has not been filled, \texttt{\textbackslash \textbackslash \texttt{metaget}} returns nothing.

\texttt{\textbackslash \texttt{metaif}} Sometimes one may want to test whether a register is filled or not, e.g. in order to display a default value otherwise. This can be achieved by the conditional:

\texttt{\textbackslash \textbackslash \texttt{metaif}\{reg\}\{true\}\{false\}}

If the register \texttt{reg} is filled, return \texttt{true} otherwise \texttt{false}. Again, the argument in square brackets is mandatory.

\texttt{\textbackslash \textbackslash \texttt{metaunset}} The following command clears a register \texttt{reg} which has previously been filled:

\texttt{\textbackslash \textbackslash \texttt{metaunset}\{reg\}}

Note that cleaning is different from filling an empty string when it comes to the conditional \texttt{\textbackslash \textbackslash \texttt{metaif}} which evaluates true for an empty string but false for a clean register.

\texttt{\textbackslash \textbackslash \texttt{metaappend}\ \textbackslash \textbackslash \texttt{metaprepend}\ \textbackslash \textbackslash \texttt{metaaddsep}} The content of registers can be manipulated by some commands. To append or prepend a string to a register, use the commands:

\texttt{\textbackslash \textbackslash \texttt{metaappend}\{reg\}\{def\}}
\texttt{\textbackslash \textbackslash \texttt{metaprepend}\{reg\}\{def\}}
\texttt{\textbackslash \textbackslash \texttt{metaaddsep}\{reg\}\{sep\}\{def\}}

The latter command \texttt{\textbackslash \textbackslash \texttt{metaaddsep}} is designed to compose lists with separators, it appends the separator \texttt{sep} and the value \texttt{def} unless the register is clean, in which case it is set to \texttt{def} without the separator \texttt{sep}. 

3
2.3 String Variants

A versatile feature of the registers is that they can be provided in several variants. These variants can be used for producing different representations of the same register depending on the intended situation. For example, a title could be given in a fully formatted version \texttt{print} for printing, a bare version for metadata and a shortened version for headings. Similarly, translations to different languages could be stored as different versions of the same register, see section 2.6. Moreover, certain attributes related to the registers could be stored in additional variants.

For example, the register \texttt{titletext} exists in the default variant (mainly intended for writing out metadata) and the \texttt{print} variant (for printing out a combination of title data on the title page). The default variant expands to a combination of \texttt{draft}, \texttt{title} and \texttt{subtitle} (as far as filled):

\[
\text{[draft: ]title[ – subtitle]}
\]

The \texttt{print} variant is accessed by the command:

\texttt{\metapick\{print\}\{titletext\}}

It expands to four lines containing \texttt{title}, \texttt{subtitle + draft}, \texttt{author} and \texttt{location + date} (as far as filled):

\[
title
subtitle \par draft
author
location, date
\]

The formatting style of each line is given by the variant \texttt{style} of the first register on this line; the vertical space above each line is produced by the variant \texttt{skip}. Two items on a single line are separated by the variant \texttt{sep} of the second register; an unfilled \texttt{sep} variant puts the two items on individual lines (by default this applies to \texttt{subtitle} and \texttt{draft}). In this way the layout of the title display on the title page can be adjusted conveniently (by default it mimics the \LaTeX command \texttt{\maketitle} in the class \texttt{article}).

Variants are always specified by an argument \texttt{[var]} in square brackets preceding the register \{reg\}. This argument is optional for commands which set registers and which can be fragile; it is however mandatory for macros which read the register content and whose output needs to be expandable into the output stream (even though the register is not optional, it is more uniform to stick with square brackets to specify the variant). The main declaration involving variants is:

\texttt{\metaset\{var\}\{reg\}\{def\}}

This command defines the register \texttt{reg} in variant \texttt{var} as \texttt{def}. The default variant is the empty string, while the variant \texttt{print} is intended for printed output. The variant string is obtained by:

\texttt{\metaget\{var\}\{reg\}}

\texttt{\metapick} The variant mechanism can become powerful through macros which fall back to default variants if the desired variant has not been filled explicitly:

\texttt{\metapick\{var\}\{reg\}}
This macro tests whether the variant \textit{var}, the language variant \textit{ln} specified through \texttt{\textbackslash metasetlang\{ln\}}, the generic (empty) variant or a fallback language variant have been specified. If so, their value is returned (in this order of preference). Importantly, the intended variant \textit{var} is passed along to the evaluation of \textit{reg} as the argument ‘\#1’ in the definition string \texttt{\textbackslash def} of \texttt{\textbackslash metaset}. This allows to define a register in one generic variant which composes other registers in more specific variants. To that end reference registers should be accessed by the construct:

\begin{verbatim}
\metaset\{reg1\}{...\metapick[#1]{reg2}...}
\end{verbatim}

When this register is accessed by \texttt{\textbackslash metapick\{var\}\{reg1\}}, it will read the default variant of \textit{reg1} which will pass on to \textit{reg2} in \textit{var} (rather than in the default variant).

There also exist a corresponding conditional:

\begin{verbatim}
\metaifpick\{var\}\{reg\}{true}{false}
\end{verbatim}

This command tests whether any of the above variants \textit{var} have been filled. Another convenient macro to more efficiently compose strings is:

\begin{verbatim}
\metacompose\{var\}\{reg\}\{prefix\}\{postfix\}\{empty\}
\end{verbatim}

It returns the intended register value with prefix string \textit{prefix} and suffix string \textit{suffix} if any of the above variants have been filled; otherwise it returns \textit{empty}. For example, the prefix and/or suffix could be separators for displaying the content of an optional register.

In dealing with variants, the following commands specify the variant \textit{var} as an optional argument \texttt{\{var\}}:

\begin{verbatim}
\metaset\{var\}\{reg\}\{def\}
\metaunset\{var\}\{reg\}
\metaappend\{var\}\{reg\}\{def\}
\metaprepend\{var\}\{reg\}\{def\}
\metaaddsep\{var\}\{reg\}\{sep\}\{def\}
\end{verbatim}

For the following macros, specifying the variant \textit{var} as \texttt{\{var\}} is \textit{mandatory}:

\begin{verbatim}
\metaget\{var\}\{reg\}
\metapick\{var\}\{reg\}
\metaifpick\{var\}\{reg\}\{true\}\{false\}
\metacompose\{var\}\{reg\}\{prefix\}\{postfix\}
\end{verbatim}

Here, the default variant is accessed by an empty argument \textit{var}.

To illustrate a construction using variants, let us consider the above register \textit{titletext}. It is defined in the generic variant as:

\begin{verbatim}
\metaset{titletext}{%\metacompose[#1]{draft}{: }%\metapick[#1]{title}%%%%%%%%%%%%%%%%}%\metacompose[#1]{subtitle}{\metaget[sep]{subtitle}}{}
\end{verbatim}

This expands to the prefix ‘\textit{draft: ’ (if available), the main title ‘\textit{title}’ and the suffix ‘ – \textit{subtitle}’ (if available). The \textit{print} variant to output a full title for the document is defined by:
Here, the macros `\metatitleline[two]` produce a title line consisting of one or two items. The single-item version is defined as:

```latex
\def\metatitleline[#1]{% 
 \metacompose[#1]{\metaget{#1}{\begingroup\metaget{style}{#1}}{\par\endgroup}{}}
```}

If register #2 is filled, this expands to the vertical skip defined by the variant `skip` and an encapsulated paragraph of the register value in the layout defined by the variant `style`.

### 2.4 Write Document Metadata

The contents of certain registers can be written out to PDF files as metadata using the package `hyperref` and the extension `hyperxmp`. The basic metadata registers are written out by `\metawritepdfinfo` using `hyperref`. The mapping between `metast` registers and `hyperref \hypersetup` options is given by:

- `titletext` → `pdftitle`
- `author` → `pdfauthor`
- `subject` → `pdfsubject`
- `keywords` → `pdfkeywords`

Here, `titletext` is used instead of `title` to compose information from the registers `draft`, `title` and `subtitle` (as far as filled). Note that `\metawritepdfinfo` will be effective only when invoked before the contents of the first page are written out.

Auxiliary metadata is written out using `hyperxmp` by the command `\metawritepdfaux` with the mapping:

- `url` → `pdfurl`
- `source` → `pdfsource`

Some metadata must be written out sufficiently early, i.e. in the document preamble, in order to go into effect. These include the language settings, and they are written out by `\metawritepdfpreamble` with the mapping:

- `language` → `pdflang`
- `language variant [meta]` → `pdfmetalang`
- `keeppdfinfo` → `keeppdfinfo`

Note that `pdfmetalang` is a setting of `hyperxmp` and will be ignored if the package is not loaded. Furthermore, the `hyperxmp` option `keeppdfinfo` will be set unless the package option `xmppdfinfo=false` is set.

A contact can be specified within PDF files in a standardised format using `hyperxmp`. The command `\metawritepdfcontact` passes on the following registers with the mapping:
A document copyright statement, see section 2.5, is recorded within the PDF file by \texttt{metawritepdfrights} using \texttt{hyperxmp} with the mapping:

\begin{verbatim}
rightstext \rightarrow pdfcopyright
licenseurl \rightarrow pdflicenseurl
\end{verbatim}

Finally, it makes sense to write out PDF metadata automatically. This is controlled by filling or clearing certain variants \texttt{var} of the register \texttt{writepdf}:

\begin{verbatim}
\metaset[\texttt{var}]{writepdf}{}
\or\ \metaunset[\texttt{var}]{writepdf}
\end{verbatim}

If the variant \texttt{auto} is filled (default), PDF metadata is written automatically at the beginning of the \texttt{document} block by calling \texttt{metawritepdf}. The command \texttt{metawritepdf} calls the commands \texttt{metawritepdf}... depending on whether the variants \texttt{info}, \texttt{aux}, \texttt{preamble}, \texttt{contact}, \texttt{rights} of the register \texttt{writepdf} are filled; the variants \texttt{info}, \texttt{aux}, \texttt{preamble} are enabled by default, the variants \texttt{contact}, \texttt{rights} need to be enabled explicitly.

Note that the basic metadata such as \texttt{author} and \texttt{title} do not have to be defined already in the preamble, but (depending on the combination of drivers and packages) they can be set before the contents of the first page are shipped out to the PDF file. If the basic registers are to be declared on the first page, one should disable their automatic writing by \texttt{metaunset[\texttt{info}]{writepdf}}. When the corresponding registers have been filled, (but no later than the end of the first page), they need to be written manually by invoking \texttt{metawritepdinfo}.

\section{Copyright and Licenses}

Specifying a copyright statement and a license is very useful because it makes the allowed (re)use of the provided material evident to the reader. However, it also takes some efforts to set things up properly. The package \texttt{metastr} provides some default texts to state the license for a couple of well-established licenses. For instance, the set of Creative Commons licenses has become a standard to mark the intended (re)use of a document involving creative content. For documents related to software, there is a number of standard software licenses to choose from.

\subsection*{Registers.} The package declares the following registers to state the copyright:

\begin{verbatim}
copyrightmark \rightarrow \texttt{\textasciitilde\textcircled{\textless\textgreater}} (\texttt{print} variant) or \texttt{Copyright}
copyrightdate \rightarrow \texttt{copyright\ date}
copyrightowner \rightarrow \texttt{copyright\ owner}
copyrightstatement \rightarrow \texttt{combines: ...mark + ...date + ...owner}
copyrightmessage \rightarrow \texttt{a message explaining the copyright situation}
\end{verbatim}

The package declares the following registers to state the license:
Furthermore, there are some related auxiliary registers:

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>partof</td>
<td>specifies the work this document is a part of</td>
</tr>
<tr>
<td>partofmessage</td>
<td>a message declaring being part of</td>
</tr>
<tr>
<td>attributionmessage</td>
<td>a message declaring attributions</td>
</tr>
<tr>
<td>rightstext</td>
<td>a composition template for all of the above</td>
</tr>
</tbody>
</table>

The above information is compiled automatically in the register `rightstext`. It can be written as PDF metadata as well as printed with formatting:

\metapick{rightstext}

**Presets.** The package provides a couple of presets for commonly used copyright statements and licenses. These are selected by:

\metacopyright{preset}
\metalicense{preset}

The following `preset` values provide the associated copyright statements:

- **plain**: This work is protected by copyright.
- **parts**: This work as well as its parts is protected by copyright.
- **doc**: This document is protected by copyright.
- **doc-parts**: This document as well as its parts is protected by copyright.
- **reserved**: All rights reserved.
- **publicdomain**: This work is dedicated to the public domain.

The following `preset` values provide the associated license statements:

- **consent**: Reproduction of any part of this work in any form without prior written consent of the author is not permissible.
- **consent-noncom**: Reproduction of any part of this work in any form without prior written consent of the author is permissible only for private, scientific and non-commercial use.
- **lppl**: This work may be distributed and/or modified under the conditions of the LaTeX Project Public License, either version 1.3 of this license or (at your option) any later version.

http://www.latex-project.org/lppl.txt

The license URL will be selected where available. The italicised parts of the license statement can be customised by the registers `licenseversion` and `licenseprovider`.

**Creative Commons Licenses.** A Creative Commons license can be selected by the command:

\metalicensecc{license}
The parameter *license* specifies the type of CC license:

- **by** Attribution
- **by-sa** Attribution-ShareAlike
- **by-nd** Attribution-NoDerivatives
- **by-nc** Attribution-NonCommercial
- **by-nc-sa** Attribution-NonCommercial-ShareAlike
- **by-nc-nd** Attribution-NonCommercial-NoDerivatives
- **zero** CC0 public domain declaration
- **pd** generic public domain declaration

The appropriate license URL and CC logo is selected by the command as well, e.g.

![Creative Commons License](https://creativecommons.org/licenses/by-sa/4.0/)

Note that **pd** is not a CC license, but it declares that the document is in the public domain by `\metacopyright{publicdomain}` and it selects the corresponding CC logo for public domain content.

A version of the CC license can be specified by the register `licenseversion`. The default version is **4.0** (international), further available versions are **3.0** (unported) as well as **2.5**, **2.0**, **1.0** (generic). For the CC0 license **zero**, the only available version is **1.0** (universal) which is the default.

Displaying the logo requires (manual) loading of the package `graphicx`; furthermore the package `doclicense` containing the logo files must be present. The display of the logo can be disabled by the package option `cclogo=false`. The logo display is coded by the following definitions which can be customised:

```
\metaset[print]{licenselogomessage}{% 
centerline{\metapick[#1]{licenselogo}}} 
\metaset[cmd]{licenselogo}{\includegraphics{#1}}
```

Various registers and variants of the selected CC license exist. The registers specific to CC licenses are:

- **licencecc** CC license identifier
- **licenceccver** CC license version
- **licenceccfull** full license descriptor

The variants specific to CC licenses are:

- **ln** representation in language ln
- **icon** CC icon (package `ccicons` required), e.g. ☑️
- **url** license URL
- **ident** CC identifier, e.g. ‘BY-SA’
- **short** short form, e.g. ‘CC BY-SA’
- **logo** logo filename (package `doclicense`)

For example, a full license descriptor is displayed by `\metapick[]{licenceccfull}`: Creative Commons License “Attribution-ShareAlike 4.0 International”. The license icon can be displayed by `\metaget[icon]{licencecc}: ☑️` (this requires the package `ccicons`
to be loaded). Note that displaying the full license message \texttt{licensemessage} in variant \texttt{print} in some languages may produce quotation marks not declared in default fonts causing an error; this can be avoided to some extent by loading an appropriate packages for internationalisation such as \texttt{babel}.

### 2.6 Languages

A principal application of the register variants is to implement internationalisation and localisation. Evidently, this is a tricky subject due to various particularities of languages, but the register variants can be used to specify and select different language representations for some commonly used text elements. For example, the copyright and license statements in section 2.5 are internationalised (to some extent). This makes them conveniently usable in the appropriate language. Note that the language presets to be loaded need to be specified explicitly by the package option \texttt{loadlang}, see section 2.7.

The idea is to understand the variant \texttt{var} of a register \texttt{reg} to be its representation in the language \texttt{var=ln[-CN]}. The default (empty) variant as well as specific purpose variants (such as \texttt{print}) should be provided in the document language or a fallback language (such as English). The macro \texttt{\metapick[\textit{var}]{\textit{reg}}} then selects the appropriate language representation or falls back to the default language. Here \texttt{var} can specify a particular language or a particular purpose. Then, \texttt{\metapick} will pick (in this order of preference):

- the language or purpose \texttt{var},
- the document language specified by \texttt{\metasetlang} (if available),
- the document language specified by \texttt{\metasetlang} with country code stripped (if available),
- the default variant,
- the fallback language (first of package option \texttt{loadlang}).

Note that nesting of \texttt{\metapick} via \texttt{\metapick[#1]{\textit{reg}}} passes along the original variant \texttt{var} in the parameter \texttt{#1}. This mechanism allows to specify some non-specific elements in a universal language while the appropriate language is selected where available.

The package reserves registers of the form \texttt{term-term} for storing terms in various (language) representations. A couple of such term registers describing common entities in typesetting are defined by the package:

- \texttt{title} \hspace{1cm} \texttt{appendix}
- \texttt{abstract} \hspace{1cm} \texttt{page}
- \texttt{copyright} \hspace{1cm} \texttt{figure}
- \texttt{preface} \hspace{1cm} \texttt{table}
- \texttt{part} \hspace{1cm} \texttt{contents}
- \texttt{chapter} \hspace{1cm} \texttt{listfigure}
- \texttt{section} \hspace{1cm} \texttt{listtable}
- \texttt{subsection} \hspace{1cm} \texttt{references}
- \texttt{paragraph} \hspace{1cm} \texttt{index}
- \texttt{appendix} \hspace{1cm} \texttt{Page}
- \texttt{Copyright} \hspace{1cm} \texttt{Figure}
- \texttt{Preface} \hspace{1cm} \texttt{Table}
- \texttt{Part} \hspace{1cm} \texttt{Contents}
- \texttt{Chapter} \hspace{1cm} \texttt{List of Figures}
- \texttt{Section} \hspace{1cm} \texttt{List of Tables}
- \texttt{Subsection} \hspace{1cm} \texttt{References}
- \texttt{Index}
- \texttt{DRAFT}

These are provided in different languages for convenient internationalisation (this can be viewed as a low-key implementation of some of the features of the \texttt{babel} package). Additional term registers can be defined by the user. Term registers are accessed by the macros:
The macro \texttt{\textbackslash metaterm\{term\}} obtains the term \textit{term} in the default language (it invokes \texttt{\textbackslash metapick[]}) with empty variant), while \texttt{\textbackslash metatranslate} uses any other language \textit{ln}. The macro \texttt{\textbackslash metasetterm[ln\{reg\}]{def}} declares and fills a term register \textit{term} (in a particular language \textit{ln}). Note that therefore it is not necessary to declare term registers explicitly by \texttt{\textbackslash metadef}.

The PDF metadata are written out in the metadata language variant specified by \texttt{\textbackslash metasetlang*}; otherwise in the default document language specified by \texttt{\textbackslash metasetlang} is used. Some registers can even be written out in several alternative language versions using the package \texttt{hyperxmp}, namely \textit{title}, \textit{subject} and \textit{rightstext}. The set of alternative languages is specified by (before invoking the respective command \texttt{\textbackslash metawritepdf...}):

\begin{verbatim}
\metaset[altlang\{title\}]{languages}
\metaset[altlang\{subject\}]{languages}
\metaset[altlang\{rightstext\}]{languages}
\end{verbatim}

Here, \textit{languages} is a comma-separated list of language identifiers and for each identifier \textit{ln} the information is written out in the respective language variant.

### 2.7 Package Options

General options for the package can be selected by the commands:

\begin{verbatim}
\usepackage[opts]{metastr}
\end{verbatim}

\texttt{\PassOptionsToPackage{opts}{metastr}} must be used before \texttt{\usepackage}. \textit{opts} is a comma-separated list of options.

The following options are available:

- \texttt{\textbackslash hyperref\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{true}) – use the package \texttt{hyperref} to write metadata to PDF.
- \texttt{\textbackslash hyperxmp\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{true}) – use the auxiliary package \texttt{hyperxmp} to write additional metadata to PDF.
- \texttt{\textbackslash checkdef\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{true}) – check whether registers have been previously declared when filling them.
- \texttt{\textbackslash cclogo\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{true}) – display CC logo from \texttt{doclicense} package.
- \texttt{\textbackslash cclogocurr\{dollar\mid euro\mid yen\}} (initially set to \texttt{dollar}) – select currency symbol for Creative Commons NonCommercial logos.
- \texttt{\textbackslash xmppdfinfo\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{true}) – write the basic PDF info block when using the auxiliary package \texttt{hyperxmp}; if this option is set, \texttt{hyperxmp} is loaded with the option \texttt{keeppdfinfo}.
- \texttt{\textbackslash draft\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{false}) – fill \texttt{draft} register with “DRAFT”.
- \texttt{\textbackslash course\{true\mid false\}} (no value implies \texttt{true}, initially set to \texttt{false}) – Setup extended registers for course materials, see section 2.8.
• \texttt{loadlang=ln-1|ln-2|\ldots|ln-n} (bar-separated list w/o spaces, initially set to \texttt{en}) – Load presets for languages \texttt{ln-1}, \texttt{ln-2}, \ldots, \texttt{ln-n}, see section 2.6. The first language \texttt{ln-1} serves as the fallback variant. Available internationalisations currently consist of:
  
  \begin{verbatim}
  en   English
  de   German
  fr   French
  es   Spanish
  \end{verbatim}

2.8 Extras

The package can provide some special purpose registers on request.

**Course Materials.** A couple of registers for course materials are provided upon setting the package option \texttt{course}:

\begin{verbatim}
\begin{tabular}{ll}
  course & title of course \\
  material & description of document material \\
  period & period where course takes place \\
  institution & institution where course is given \\
  instructor & instructor of course \\
\end{tabular}
\end{verbatim}

The variant \texttt{course} of the register \texttt{titletext} displays a compilation of these registers for display on a title page:

\begin{verbatim}
\verb|\course| material \verb|--| draft \\
\verb|\institution|, \verb|\period|
\end{verbatim}

Furthermore, the registers \texttt{title}, \texttt{subtitle}, \texttt{author}, \texttt{location} and \texttt{date} are diverted to \texttt{course}, \texttt{material}, \texttt{instructor}, \texttt{institution} and \texttt{period} respectively. Consequently, their values are automatically written as PDF metadata, but it is certainly possible to override them with custom values.

3 Information

3.1 Copyright

Copyright © 2020 Niklas Beisert

This work may be distributed and/or modified under the conditions of the \LaTeX\ Project Public License, either version 1.3 of this license or (at your option) any later version. The latest version of this license is in \url{http://www.latex-project.org/lppl.txt} and version 1.3 or later is part of all distributions of \LaTeX\ version 2005/12/01 or later.

This work has the LPPL maintenance status ‘maintained’.

The Current Maintainer of this work is Niklas Beisert.

This work consists of the files \texttt{README.txt}, \texttt{metastr.ins} and \texttt{metastr.dtx} as well as the derived files \texttt{metastr.sty}, \texttt{metasamp.tex} and \texttt{metastr.pdf}. 

12
3.2 Files and Installation

The package consists of the files:

- README.txt: readme file
- metastr.ins: installation file
- metastr.dtx: source file
- metastr.sty: package file
- metasamp.tex: sample file
- metastr.pdf: manual

The distribution consists of the files README.txt, metastr.ins and metastr.dtx.

- Run (pdf)\LaTeX\ on metastr.dtx to compile the manual metastr.pdf (this file).
- Run \LaTeX\ on metastr.ins to create the package metastr.sty and the samples consisting of metasamp.tex. Copy the file metastr.sty to an appropriate directory of your \LaTeX\ distribution, e.g. texmf-root/tex/latex/metastr.

3.3 Related Packages

The package makes use of other packages available at CTAN:

- This package uses the package hyperref to write basic metadata to a PDF file. Compatibility with the hyperref package has been tested with v7.00c (2019/11/10).
- This package uses the package keyval from the graphics bundle to process the options for the package, environments and macros. Compatibility with the keyval package has been tested with v1.15 (2014/10/28).
- This package can use the package hyperxmp to write extended metadata to a PDF file. Compatibility with the hyperxmp package has been tested with v4.1 (2019/04/05).
- This package can use the Creative Commons license icon files included in the package doclicense. Compatibility with the doclicense package has been tested with v1.10.0 (2019/06/05).
- This package can use the Creative Commons license icon fonts included in the package ccicons. Compatibility with the ccicons package has been tested with v1.6 (2017/10/30).
- Icon files are displayed by means of the graphicx package. The package needs to be loaded explicitly. Compatibility with the graphicx package has been tested with v1.1a (2017/06/01).

There are several other \LaTeX\ packages which store and write basic metadata for some specific purposes:

- The package hyperref writes the arguments of \author\ and \title\ unless the package option pdfusetitle=false is declared (at load time).
- The package hyperxmp writes the arguments of \author\ and \title.
- The package exframe writes the \exercisedata\ registers author, title, subject and keyword unless the package option pdfdata=off is specified.
- The package beamer writes the arguments of \author, \title, \subject and \keywords.
- The package gitver writes pdfsubject unless the package option nopdfinfo is specified.
Various packages to prepare articles for publication in journals. Their mechanisms may be in competition with the ones of the present package \texttt{metastr}. In order to make the packages work together on the same set of data, the most promising option which should work in many cases is the following: Fill the registers of \texttt{metastr} with the desired values. Then pass them on to the structures of the other package(s) using \texttt{metaget} or \texttt{metapick}. Since the latter commands are robust, the other structures ought to be able to handle them without further ado. To avoid potential conflicts, multiple writing of (basic) metadata should be disabled. For the \texttt{metastr} package this is achieved by:

\begin{verbatim}
\metaunset[info]{writepdf}
or \metaunset[auto]{writepdf}
\end{verbatim}

3.4 Feature Suggestions

The following is a list of features which may be useful for future versions of this package:

- Presets for GNU and other software licenses.
- Registers for publication data.
- Further translations of copyright and license statements.
- Export translations to files
- Make use of the \texttt{babel} package for translations of basic terms.

3.5 Revision History

\textbf{v1.0:} 2020/02/06

- first version, published on CTAN

A Sample

This section provides an example of how to apply some of the \texttt{metastr} mechanisms and licenses.

Some lines in the example are commented by \texttt{%%} for easy experimenting.

\begin{verbatim}
\documentclass[12pt]{article}
\usepackage{geometry}
\geometry{layout=a4paper}
\geometry{paper=a4paper}
\geometry{margin=2.5cm}
\parindent0pt
\parskip1ex
\PassOptionsToPackage{bookmarks=true}{hyperref}
\usepackage{hyperref}
\end{verbatim}

Preamble. Standard document class:

\begin{verbatim}
\documentclass[12pt]{article}
\usepackage{geometry}
\geometry{layout=a4paper}
\geometry{paper=a4paper}
\geometry{margin=2.5cm}
\parindent0pt
\parskip1ex
\PassOptionsToPackage{bookmarks=true}{hyperref}
\usepackage{hyperref}
\end{verbatim}

Declare some options for the package \texttt{hyperref}; it does not hurt to load it explicitly although \texttt{metastr} will invoke it by default if not loaded:

\begin{verbatim}
\PassOptionsToPackage{bookmarks=true}{hyperref}
\usepackage{hyperref}
\end{verbatim}
Set some options for the `metastr` package:

10 \PassOptionsToPackage{draft}{metastr}
11 \PassOptionsToPackage{loadlang=en|de|fr|es}{metastr}
12 \PassOptionsToPackage{loadlang=en|fr|es}{metastr}
13 \PassOptionsToPackage{loadlang=de|en}{metastr}
14 \PassOptionsToPackage{cclogocurr=euro}{metastr}

Include the `metastr` package along with `graphicx`, `babel` and `ccicons` (where available):

15 \usepackage{metastr}
16 \usepackage{graphicx}
17 \usepackage{babel}
18 \IfFileExists{ccicons.sty}{\usepackage{ccicons}}{}

Some Adjustments. Declare some term to be translated; doesn’t hurt to declare a couple of variants:

19 \metasetterm[en]{Zurich}{Zurich}
20 \metasetterm[de]{Zurich}{Z"rich}
21 \metasetterm[fr]{Zurich}{Zurich}
22 \metasetterm[es]{Zurich}{Z"urich}
23 \metasetterm[it]{Zurich}{Zurigo}
24 \metasetterm[pt]{Zurich}{Zurique}

Define subject to combine location and date (as far as filled):

25 {\metacompose[#1]{location}{location: }}
26 {\metacompose[#1]{date}{, date: }{}{}}
27 {\metacompose[#1]{date}{date: }{}{}}

Adjust title display:

28 \metaset{skip}{subtitle}{\vspace{1ex}}
29 \metaset{skip}{author}{\vspace{2ex}}
30 \metaset{skip}{location}{\vspace{1ex}}
31 \metaset{skip}{date}{\vspace{1ex}}
32 \metaset{style}{title}{\LARGE\bfseries}
33 \metaset{style}{author}{\large\scshape}
34 \metaset{sep}{draft}{ -- }
35 \metaset{sep}{date}

Write title also in english and german; write rights as PDF metadata also in English and Spanish:

36 \metaset[altlang]{title}{en,de}
37 \metaset[altlang]{rightstext}{en,es}
38 \metaset[rights]{writepdf}{}

Set Document Data. Set the document language:

39 \metasetlang{en}
40 \metasetlang{de}
41 \metasetlang{de-CH}
42 \metasetlang{fr}

Define some document data:

43 \metaset[en]{title}{A metastr Sample}
44 \metaset[de]{title}{Ein metastr Beispiel}
45 \metaset[print]{title}{A \textsf{metastr} Sample}
Header. Display title block:

\begin{document}
\begin{center}
\metapick{print}{titletext}
\end{center}
\vspace{1ex}\hrule\par\vspace{1ex}
\begingroup\footnotesize
\pdfbookmark{\metaterm{copyright}}{copyright}
\metapick{print}{rightstext}
\endgroup
\vspace{1ex}\hrule\par\vspace{1ex}

Content. Some useful content:

\section{Metadata Inspection}

The metadata stored in this example PDF can be inspected with the tool \verb+pdfinfo+:
Translations. Demonstration of terms and translations:

\begin{tabular}{ll}
document language:&\metaterm{Zurich}\\
Spanish:&\metatranslate[es]{Zurich}
\end{tabular}

Creative Commons. Demonstrate some CC terms:

\metaif\{cc@type\}{ % only if a CC license is in use
\section{Creative Commons}
some representations of the selected license:
\begin{itemize}
\item license identifier:
\metapick\{licensecc\}
\item \texttt{short} identifier:
\metapick[short]\{licensecc\}
\item full form:
\metapick\{licenseccfull\}
\item \texttt{ident} form:
\metapick[ident]\{licensecc\}
\item \texttt{short} form:
\metapick[short]\{licenseccfull\}
\IfFileExists{ccicons.sty}
\item \texttt{icon} forms:
-- \metapick[icon]\{licensecc\}
-- \metapick[icon]\{licenseccfull\} --}
\item \texttt{url} form:
\metapick[url]\{licenseccfull\}
\end{itemize}
\end{document}

B Implementation

This section describes the implementation of the package metastr.sty.

B.1 Package Setup

The package declares a couple of setup options.
It loads the package keyval for extended options processing.
\RequirePackage{keyval}

Store the selected package options in some corresponding internal macros:
\newif\ifmstr@opt@hyperref\mstr@opt@hyperreftrue
Pass undeclared options on to keyval processing:

\PassOptionsTo{\keyval}{\CurrentOption}

Pass global options while loading package:

\ProcessOptions

### B.2 Definitions

The following describes the basic definitions of the package.

**Required Packages.** The package loads the packages `hyperref` and `hyperxmp` (unless excluded):

\ifmstr@opt@hyperref\RequirePackage{hyperref}\fi
\ifmstr@opt@hyperxmp\RequirePackage{hyperxmp}\fi

**General Definitions.**

\mstr@exptwo A macro to conveniently expand the third token in line:

\def\mstr@exptwo#1\expandafter#1\expandafter

\mstr@csdo Some macros to conveniently expand `\csname` arguments before expanding the macro:

\mstr@csdotwo

\mstr@iftext Check whether macro #1 equals text #2, then do #3:

\longdef\mstr@iftext#1#2#3\def\mstr@tmp{#2}\ifx#1\mstr@tmp#3\fi
Internal Definitions.

\set\language@main@empty
\set\language@short@empty
\set\language@fallback@empty
\set\language@meta@empty

Interface Definitions.

\set@tilde
\set@comma

Define a macro for the tilde character (mostly for use within URLs); recycle the definitions
from hyperxmp if available:

\ifdefined\xmtt\let\set@tilde\xmtt\let\set@comma\xmpcomma\else\def\set@tilde{~}\def\set@comma{,}\fi

Declare Registers.

\set@def
Declare a register:

\newcommand{\set@def}[1]{\let\set@def@#1\relax}

\set@verify
Verify the declaration of a register; throw an error if undeclared; disable checking for package
option \texttt{checkdef=false}:

\newcommand{\set@verify}[1]{\ifcsname \set@def@#1\endcsname\else
\PackageError{metastr}{register '\#1' undefined}{}\fi\if\set@opt@checkdef\else\def\set@verify#1\fi}

Set Registers.

\set@bare
Store the register value in the macro \texttt{\set@data@reg@var}; define one argument to pass
along original variant:

\long\def\set@bare[#1]{\let\set@data@#1\relax}

\set@set
Set the register value; verify whether the register has been declared:

\long\def\set@set[#1]{\set@verify{#1}\set@bare[#1]{#2}{#3}}

\set@set
Interface macro for setting register with optional variant argument:

\newcommand{\set@set}{\@ifnextchar[{{\set@set}{\set@set[]}}

19
\mstr@unset Clear a register value:

\begin{verbatim}
170 \long\def\mstr@unset[#1]{\mstr@verify[#2]\
171 \mstr@csdotwo\global\let\mstr@data@#2@#1}\@undefined
\end{verbatim}

\metaunset Interface macro for clearing register with optional variant argument:

\begin{verbatim}
172 \newcommand{\metaunset}{\@ifnextchar[{{\mstr@unset}{\mstr@unset[{}\]
\end{verbatim}

Register Conditionals.

\metaif If-then-else structure checking whether register variant is filled:

\begin{verbatim}
173 \long\def\metaif[#1]{#2#3#4{\
174 \ifcsname mstr@data@#2@#1\endcsname #3\else #4\fi}
\end{verbatim}

\metaifpick If-then-else structure checking if the register in either of the variants #1, \mstr@lang@main, \mstr@lang@short, default and \mstr@lang@fallback is filled; #1 may in fact be a comma-separated list of variants (without spaces):

\begin{verbatim}
175 \long\def\mstr@ifloop[#1,#2]{#2{#3}#4{if @#2@#5else mstr@ifloop[#2]{#3}{#4}{#5}fi}}
177 \long\def\metaifpick[#1]{#2#3#4{\
178 mstr@ifloop
179 [#1,\mstr@lang@main,\mstr@lang@short,,\mstr@lang@fallback,]
180 {#2}{#3}{#4}}
\end{verbatim}

Manipulate Registers.

\mstr@append Append some string to a register value:

\begin{verbatim}
181 \long\def\mstr@append[#1]{#2#3{\
182 \mstr@csdotwo\let\mstr@tmpa{mstr@data@#2@#1}\
183 \def\mstr@tmpb##1{\mstr@set[#1]{#2}{##1#3}}\
184 \mstr@exptwo\mstr@tmpb{\mstr@tmpa{##1}}}
\end{verbatim}

\mstr@prepend Prepend some string to a register value:

\begin{verbatim}
185 \long\def\mstr@prepend[#1]{#2#3{\
186 \mstr@csdotwo\let\mstr@tmpa{mstr@data@#2@#1}\
187 \def\mstr@tmpb##1{\mstr@set[#1]{#2}{##1#3}}\
188 \mstr@exptwo\mstr@tmpb{\mstr@tmpa{##1}}}
\end{verbatim}

\mstr@addsep Append a string to a register value separated by #1 if the string was previously filled:

\begin{verbatim}
189 \long\def\mstr@addsep[#1]{#2#3#4{\
190 \metaif[#1]{#2}{\mstr@append[#1]{#2}{#3#4}}{\mstr@set[#1]{#2}{#4}}}
\end{verbatim}

\metaappend \metaprepend \metaaddsep Interface macros for appending, prepending and adding with separator:

\begin{verbatim}
191 \newcommand{\metaappend}{\
192 \@ifnextchar[{{\mstr@append}{\mstr@append[{}\]
193 \newcommand{\metaprepend}{\
194 \@ifnextchar[{{\mstr@prepend}{\mstr@prepend[{}\]
195 \newcommand{\metaaddsep}{\
196 \@ifnextchar[{{\mstr@addsep}{\mstr@addsep[{}\]}
\end{verbatim}
Read Register Values.

\mstr@getbare Read a register value while passing along the original variant as an argument:

\begin{verbatim}
197 \def\mstr@getbare[#1]{\csname mstr@data@#2@#1\endcsname{#3}}
\end{verbatim}

\metaget Interface function to read register value with mandatory variant argument in square brackets; return nothing if register clean:

\begin{verbatim}
198 \def\metaget[#1]{#2}{\mstr@getbare[#1]{#2}{#1}}%
\end{verbatim}

\metacompose \metapick returns a filled register value among the variants #1, \mstr@lang@main, \mstr@lang@short, default and \mstr@lang@fallback (in this order of preference), otherwise it returns nothing; #1 may in fact be a comma-separated list of variants (without spaces); \metacompose sandwiches the value between #3 and #4 if found, and otherwise returns #5:

\begin{verbatim}
200 \long\def\mstr@composeloop[#1,#2,#3,#4,#5,#6,#7]{% 
201 \metaif[#1]{#4}{#5\mstr@getbare[#1]{#4}{#3}#6}{\if @#2@#7\else\mstr@composeloop[#2]{#3}{#4}{#5}{#6}{#7}\fi}}
203 \long\def\metacompose[#1]{#2}{#3}{#4}{#5}{#6}{#7}{}
\end{verbatim}

\metapick language Declare language register:

\begin{verbatim}
208 \metadef{language}
\end{verbatim}

\metasetlang Set language and extract short forms:

\begin{verbatim}
209 \def\mstr@lang@split#1#2-#3@{\mstr@csdo\gdef{mstr@lang@#1}{#2}}
212 \metaset{language}{#1}{}
213 \gdef\mstr@lang@main{#1}{}
214 \mstr@lang@split{short}#1-@{}% 
216 \metaif[meta]{language}{}{\mstr@lang@split{meta}#1-@}{% 
217 \newcommand{\mstr@setlang@meta}{#1}{}% 
218 \metaif[meta]{language}{}{\mstr@lang@split{meta}#1-@}{% 
220 \newcommand{\metasetterm}{\@ifnextchar[\mstr@setterm}{\mstr@setterm[]}}
222 \@ifstar{\mstr@setlang@meta}{\mstr@setlang@main}
\end{verbatim}

Terms.

\metaterm \metatranslate \metasetterm Macros for filling and reading term registers:

\begin{verbatim}
224 \def\metatranslate[#1]{\metapick[#1]{term-#2}}
225 \long\def\mstr@setterterm[1]{#2}{\mstr@setterterm[2]{#3}}% 
227 \newcommand{\metasetterm}{\@ifnextchar{\mstr@setterterm}{\mstr@setterterm[]}}
\end{verbatim}
Automatic Writing to PDF.

\texttt{writepdf} Declare register \texttt{writepdf} to control automatic writing of metadata to PDF files:

\begin{verbatim}
228 \metadef{writepdf}
229 \metaset[auto]{writepdf}{}
230 \metaset[preamble]{writepdf}{}
231 \metaset[info]{writepdf}{}
232 \metaset[aux]{writepdf}{}
\end{verbatim}

\texttt{mstr@ifwritepdf} Auxiliary macro to write some type of metadata if switch activated, disable switch afterwards:

\begin{verbatim}
233 \longdef\mstr@ifwritepdf[#1]#2{%
234 \metaif[#1]{writepdf}{#2\metaunset[#1]{writepdf}}{}}
\end{verbatim}

\texttt{metawritepdf} Write selected types of metadata to PDF file:

\begin{verbatim}
235 \newcommand{\metawritepdf}{%\mstr@ifwritepdf[preamble]{\metawritepdfpreamble}%\mstr@ifwritepdf[info]{\metawritepdfinfo}%\mstr@ifwritepdf[aux]{\metawritepdfaux}%\mstr@ifwritepdf[contact]{\metawritepdfcontact}%\mstr@ifwritepdf[rights]{\metawritepdfrights}%
\end{verbatim}

\texttt{mstr@begindoc} Hook for writing data to PDF file; this is the last chance to write the preamble set of data to the PDF:

\begin{verbatim}
242 \newcommand{\mstr@begindoc}{%\mstr@ifwritepdf[preamble]{\metawritepdfpreamble}%\mstr@ifwritepdf[auto]{\metawritepdf}%
\end{verbatim}

B.3 Basic Registers

The following defines a set of basic and auxiliary registers.

Declarations.

\texttt{draft} Declare register to state draft mode:

\begin{verbatim}
250 \metadef{draft}
\end{verbatim}

Set draft text (if \texttt{draft} option set):

\begin{verbatim}
251 \ifmstr@opt@draft
252 \metaset{draft}{\metatranslate[#1]{draft}}
253 \fi
\end{verbatim}

\texttt{title} \texttt{subtitle} \texttt{author} \texttt{date} \texttt{location} \texttt{subject} \texttt{keywords}

\clearpage
Title Composition.

\titletext Declare register to compose title display (analogous to \texttt{\maketitle})

\metatitleline Macros to print a formatted title line with one or two items; variant \texttt{skip} produces vertical skip before the item, variant \texttt{style} sets the text style, variant \texttt{sep} defines the separator between two items or undefined for two independent lines:

\begin{verbatim}
\def\metatitleline[#1]{% 
\metacompose[#1]{}{\metaget[skip]{}% 
\begingroup\metaget[style]{}% 
\par\endgroup}{}}
\def\metatitlelinetwo[#1]{% 
\metacompose[#1]{\metaget[skip]{}% 
\begingroup\metaget[style]{}% 
\par\endgroup}{} \metatitleline[#3]{}}
\end{verbatim}

Set default layout and spacing:

\begin{verbatim}
\metaset[style]{title}{\LARGE}
\metaset[style]{subtitle}{\large}
\metaset[style]{draft}{\large}
\metaset[style]{author}{\large}
\metaset[style]{location}{\large}
\metaset[style]{date}{\large}
\metaset[skip]{subtitle}{\vspace{1.5em}}
\metaset[skip]{draft}{\vspace{1.5em}}
\metaset[skip]{author}{\vspace{3em}}
\metaset[skip]{location}{\vspace{1.5em}}
\metaset[skip]{date}{\vspace{1.5em}}
\metaset[sep]{subtitle}{ -- }
\metaset[sep]{date}{, }
\end{verbatim}

Preset for \texttt{titletext} in generic and print variants:

\begin{verbatim}
\metaset{titletext}{% 
\metacompose[#1]{draft}{\{}% 
\metapick[#1]{title}{% 
\metacompose[#1]{subtitle}{\metaget[sep]{subtitle}}{}% 
\}}% 
\metaset{print}{titletext}{% 
\metatitleline[print]{title}{% 
\metatitlelinetwo[print]{subtitle}[print]{draft}{% 
\metatitleline[print]{author}{% 
\metatitlelinetwo[print]{location}[print]{date}{}}}
\end{verbatim}

Further Registers.
url Registers for document URL and message to display it:

\metadef{url}
\metadef{urlmessage}

Print URL as hyperlink:

\metaset[print]{url}{\url{\metaget[]{url}}}

URL message default text (translated):

\metaset{urlmessage}{
% The current version of this work can be found at: \metapick[#1]{url}.}

partof Registers for document URL and message to display it:

\metadef{partof}
\metadef{partofmessage}

part of message default text (translated):

\metaset{partofmessage}{
% This document is part of the work: \metapick[#1]{partof}.}

source Register for source name:

\metadef{source}

Write to PDF.

\metawritepdfpreamble Write some registers to PDF that need to be written before the start of the document:

\newcommand{\metawritepdfpreamble}{\ifdefined\hypersetup
\ifmstr@opt@xmppdfinfo
\ifdefined\xmptilde\hypersetup{keeppdfinfo}\fi\fi
\metaif{}{language}
{\hypersetup{pdflang={\metaget{}{language}}}\%}
\ifdefined\xmptilde
\metaif{meta}{language}
{\hypersetup{pdfmetalang={\metaget{meta}{language}}}\%}
\fi\fi}

\metawritepdfinfo Write the basic registers to PDF; also write alternative language representations of pdftitle and pdfsubject:

\newcommand{\metawritepdfinfo}{\ifdefined\hypersetup
\metaifpick{mstr@lang@meta}{author}
{\hypersetup{pdfauthor={\metapick{mstr@lang@meta}{author}}}\%}
\metaifpick{mstr@lang@meta}{title}
{\hypersetup{pdftitle={\metapick{mstr@lang@meta}{titletext}}}\%}
\metaifpick{mstr@lang@meta}{subject}
{\hypersetup{pdfsubject={\metapick{mstr@lang@meta}{subject}}}\%}
\metaifpick{mstr@lang@meta}{keywords}
{\hypersetup{pdfkeywords={\metapick{mstr@lang@meta}{keywords}}}\%}
\ifdefined\xmptilde
\metaif{altlang}{title}{
\for\mstr@tmp:=\mstr@data@title@altlang{\do{\metaifpick{mstr@tmp}{title}}}
\XMPLangAlt{\mstr@tmp}{pdftitle}=\XMPLangAlt{\mstr@tmp}{pdfsubject}
B.4 Copyright and License

The following defines some registers concerning copyright and licensing.

Rights Composition.

rightstext Declare a register to compose copyright and license information:

\metadef{rightstext}

Define generic version of composition register:

\metaset[rightstext]{% 
\metaif{partof}{\metacompose[#1]{partofmessage}{ }{}}{}% 
\metaif{copyrightstatement}{% 
\metacompose[#1]{copyrightmessage}{ }{}% 
\metacompose[#1]{licensemessage}{ }{}% 
\metaif{licenseurl}{\metacompose[#1]{licenseurlmessage}{ }{}{}}{}% 
\metaif{url}{\metacompose[#1]{urlmessage}{ }{}{}}{}% 
\attributionmessage{}}{}}% 

Define print version of composition register; variant \texttt{sep} contains code to separate parts of the message:

\metaset[print]{rightstext}{% 
\par\addvspace\medskipamount 
\attributionmessage{}}{}}% 

\metaset[skip]{rightstext}{% 
\par\addvspace\medskipamount 
\attributionmessage{}}{}}%
Copyright Composition.

**copyright...** Declare registers to specify document copyright:

369 \metadef{copyrightmark}
370 \metadef{copyrightdate}
371 \metadef{copyrightowner}
372 \metadef{copyrightstatement}
373 \metadef{copyrightmessage}

**copyrightmark** The copyright sign or word:

374 \metaset{copyrightmark}{Copyright}
375 \metaset{print}{copyrightmark}{\copyright}

**copyrightstatement** Assemble the copyright statement from available fragments; proper spacing makes this a bit tedious:

376 \metaset{copyrightstatement}{\metaifpick\[#1\]{copyrightdate}{\metapick\[#1\]{copyrightmark} \metapick\[#1\]{copyrightdate}{\{}\{}\} \metaifpick\[#1\]{copyrightowner}{\{}\} }{}
379 \{}\}

License Composition.

**license...** Declare registers to specify document license:

381 \metadef{licenseversion}
382 \metadef{licenseprovider}
383 \metadef{licensemessage}
384 \metadef{licenselogo}
385 \metadef{licenselogomessage}
386 \metadef{licenseurl}
387 \metadef{licenseurlmessage}

**attributionmessage** Declare a register for the attribution message:

388 \metadef{attributionmessage}

**licenseurlmessage** Message to declare URL at which the relevant license or further details can be found (translated):

389 % \metaset{licenseurlmessage}{% 390 % To view a copy of this license, visit: \metapick[#1]{licenseurl}.}

**licenseurl** In \texttt{print} version, pass plain \texttt{licenseurl} through \texttt{url}:

391 \metaset{print}{licenseurl}{url\metaget[]{licenseurl}}

**licenselogo** Display license logo, by default align centrally; abuse the variant argument for passing the file name argument to \texttt{\includegraphics}:

392 \metaset{print}{licenselogomessage}{% 393 \centerline{\metapick[#1]{licenselogo}}
394 \metaset{cmd}{licenselogo}{\includegraphics[#1]}
395 \metaset{print}{licenselogo}{% 396 \make@getbare{cmd}{licenselogo}\metaget[]{licenselogo}}
Write to PDF.

Write rights information (rights text, alternative language representations, license url) to PDF via hyperxmp:

\newcommand{writepdfrights}{\ifdefined\hypersetup\ifdefined\xmpltilde
\metaifpick{\mstr@lang@meta}{rightstext}{\hypersetup{pdfcopyright={\metapick{\mstr@lang@meta}{rightstext}}}{}%\metaif{altlang}{rightstext}{\@for\mstr@tmp:=\mstr@data@rightstext@altlang{}\do\XMPLangAlt{\mstr@tmp}{pdfcopyright={\metapick{\mstr@tmp}{rightstext}}}{}\metaifpick{\mstr@lang@meta}{licenseurl}{\hypersetup{pdflicenseurl={\metapick{\mstr@lang@meta}{licenseurl}}}{}%\fi\fi}}

Copyright Presets.

copyright@... Declare some copyright presets:

\metadef{copyright@plain}
\metadef{copyright@parts}
\metadef{copyright@doc}
\metadef{copyright@doc-parts}
\metadef{copyright@reserved}
\metadef{copyright@publicdomain}

plain parts doc doc-parts reserved Some plain copyright messages (translated):

% \metaset{copyright@plain}{% This work is protected by copyright.}%
% \metaset{copyright@parts}{% This work as well as its parts is protected by copyright.}%
% \metaset{copyright@doc}{% This document is protected by copyright.}%
% \metaset{copyright@doc-parts}{% This document as well as its parts is protected by copyright.}%
% \metaset{copyright@reserved}{All rights reserved.}%

publicdomain A public domain declaration (translated):

% \metaset{copyright@publicdomain}{% This work is dedicated to the public domain.}%

License Presets.

license@... Declare some license presets

\metadef{license@consent}
\metadef{license@consent-noncom}
\metadef{license@lppl}

consent A license to reproduce with prior written consent (translated):

% \metaset{license@consent}{% Reproduction of any part of this work in any form}

27
without prior written consent
\metacompose[#1]{licenseprovider}{}{}{of the author}
is not permissible.)

\begin{itemize}
\item \textbf{consent-noncom} A license to reproduce for private, scientific and non-commercial purposes or with prior written consent (translated):
\begin{verbatim}
\metaset{license@consent-noncom}{% Reproduction of any part of this work in any form \without prior written consent \metapick[#1]{licenseprovider}{}{}{of the author} is permissible only for private, scientific and non-commercial use.}
\end{verbatim}
\end{itemize}

\begin{itemize}
\item \textbf{lppl} \LaTeX{} project public license (translated):
\begin{verbatim}
\metaset[url]{license@lppl}{http://www.latex-project.org/lppl.txt}
\metaset{license@lppl}{% This work may be distributed and/or modified under the \conditions of the \LaTeX{} Project Public License, either version \metaif{}{licenseversion}{}{\metaget{}{licenseversion}}{1.3} \% of this license or (at your option) any later version.}
\end{verbatim}
\end{itemize}

\begin{itemize}
\item \textbf{Selection Code.}
\begin{verbatim}
\newcommand{\metacopyright}[1]{% \metaset{copyrightmessage}{\metapick[#1]{copyright@#1}}}
\newcommand{\metalicense}[1]{% \metaset{licensemessage}{\metapick[#1]{license@#1}}\metaif{}{license@#1}{}{}}
\end{verbatim}
\end{itemize}

\section*{B.5 Creative Commons}
The following implements the scheme of Creative Commons licenses.

\begin{itemize}
\item \textbf{Declarations.}
\begin{verbatim}
\metadef{cc@type}
\metadef{cc@class}
\end{verbatim}
\end{itemize}

\begin{itemize}
\item \textbf{cc@type} \textbf{cc@type} stores the selected CC license type; \textbf{cc@class} is ‘\@zero’ for the CC0 public domain dedication and empty otherwise:
\begin{verbatim}
\metadef{cc@type}
\metadef{cc@class}
\end{verbatim}
\end{itemize}

\begin{itemize}
\item \textbf{Text Components and Internationalisation.} CC license declarations are composed from several elements which can be conveniently internationalised.

The following registers store various terms used in CC licenses:
\begin{verbatim}
\metasetterm{cc@sep}{-}
\metasetterm{cc@quotel}{\textquotedblleft}
\metasetterm{cc@quoter}{\textquotedblright}
\metasetterm{cc@cc}{Creative Commons}
\metasetterm{cc@by}{Attribution}
\metasetterm{cc@sa}{ShareAlike}
\end{verbatim}
\end{itemize}
The following template registers store the combinations for the various CC licenses:

\metadef{cc@pd}
\metadef{cc@zero}
\metadef{cc@by...
\metadef{cc@by-sa}
\metadef{cc@by-nd}
\metadef{cc@by-nc}
\metadef{cc@by-nc-sa}
\metadef{cc@by-nc-nd}

Fill the registers:

\metaset{cc@zero}{\metatranslate{cc@zero}}
\metaset{cc@by}{\metatranslate{cc@by}}
\metaset{cc@by-sa}{\metatranslate{cc@by}\metatranslate{cc@sa}}
\metaset{cc@by-nd}{\metatranslate{cc@by}\metatranslate{cc@nd}}
\metaset{cc@by-nc}{\metatranslate{cc@by}\metatranslate{cc@nc}}
\metaset{cc@by-nc-sa}{\metatranslate{cc@by}\metatranslate{cc@nc}\metatranslate{cc@sa}}
\metaset{cc@by-nc-nd}{\metatranslate{cc@by}\metatranslate{cc@nc}\metatranslate{cc@nd}}

The following registers store the various versions for CC licenses:

\metadef{cc@1.0@zero}
\metadef{cc@1.0}
\metadef{cc@2.0}
\metadef{cc@2.5}
\metadef{cc@3.0}
\metadef{cc@4.0}

Fill the registers:

\metaset{cc@1.0@zero}{\metatranslate{cc@univ}}
\metaset{cc@1.0}{\metatranslate{cc@generic}}
\metaset{cc@2.0}{\metatranslate{cc@generic}}
\metaset{cc@2.5}{\metatranslate{cc@generic}}
\metaset{cc@3.0}{\metatranslate{cc@unported}}
\metaset{cc@4.0}{\metatranslate{cc@intl}}
The following registers store the term “CC license”:

\metadef{cc@license}
\metadef{cc@license@zero}
\metadef{cc@license@pd}

Fill the registers (translated):

\% \metaset{cc@license}{%
\% \metatranslate\[#1\]{cc@cc} \metatranslate\[#1\]{cc@license}}
\% \metaset{cc@license@zero}{%
\% \metatranslate\[#1\]{cc@cc} \metatranslate\[#1\]{cc@pddecl}}
\metaset{cc@license@pd}{\metatranslate\[#1\]{cc@pddecl}}

The following registers contain presets for the CC license messages:

\metadef{cc@message}
\metadef{cc@message@zero}

Fill the registers (translated):

\% \metaset{cc@message}{%
\% This work is licensed under the
\% \metapick\[#1\]{licenseccfull} (\metapick[short]{licenseccfull}).}
\% \metaset{cc@message@zero}{%
\% This work is dedicated to the public domain by means of the
\% \metapick\[#1\]{licenseccfull} (\metapick[short]{licenseccfull}).}

licensecc and licenseccver represent the name and version of the selected CC license;
licenseccfull contains a full representation of the selected CC license:

\metadef{licensecc}
\metadef{licenseccver}
\metadef{licenseccfull}

Fill the registers:

\metaset{licensecc}{%
\metapick\[#1\]{cc@metaget[]}{cc@type}}
\metaset{licenseccver}{%
\metaget[]{licenseversion}}
\metaset{licenseccfull}{%
\metapick\[#1\]{cc@license}\metaget[]{cc@class}}
\metatranslate\[#1\]{cc@quotel}
\metapick\[#1\]{licensecc}
\metapick\[#1\]{licenseccver}
\metatranslate\[#1\]{cc@quoter}

License Identifier.

ident Compose the license identifier by ident variant:

\metaset[ident]{licenseccver}{\metaget[]{licenseversion}}
\metaset[ident]{licenseccfull}{%}
\metapick[ident]{licensecc} \metaget[ident]{licenseccver}
\metasetterm[ident]{cc@sep}{-}
\metasetterm[ident]{cc@cc}{CC}
\metasetterm[ident]{cc@by}{BY}
\metasetterm[ident]{cc@sa}{SA}
Compose the short license identifier by short variant:

\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}
\short\licensecc{\%}

License Logo.

\licensecc
\licensecc
\licensecc
\licensecc
\licensecc

Use euro sign versions:

\licensecc
\licensecc
\licensecc
\licensecc

Use the Creative Commons logos included in the doclicense package:

\mstr@setcclogo

icon

\licensecc

Use the Creative Commons logos included in the ccicons package:

\licensecc

\licensecc

\licensecc

\licensecc

\licensecc

\licensecc

\licensecc
License URL.

url  The url variant of compounds are used for the license URL.

Declare registers for internationalisation of deed URL:

Fill registers:

License Preset.

Set the CC license of type ‘#1’:
B.6 Contact Information

The following describes an interface to store and write contact information.

Declarations.

contact... Contact register declarations:

- \metadef{contactaddress}
- \metadef{contactpostcode}
- \metadef{contactcity}
- \metadef{contactregion}
- \metadef{contactcountry}
- \metadef{contactemail}
- \metadef{contacturl}

Write to PDF.

Write contact information to PDF via hyperxmp:

- \newcommand{\metawritepdfcontact}{\ifdefined\hypersetup\ifdefined\xmp tilted
  \metaifpick[\mstr@lang@meta]{contactaddress}{%}
  \hypersetup{pdfcontactaddress=
    \metapick[\mstr@lang@meta]{contactaddress}}{}

- \metaifpick[\mstr@lang@meta]{contactpostcode}{%}
  \hypersetup{pdfcontactpostcode=
    \metapick[\mstr@lang@meta]{contactpostcode}}{}

- \metaifpick[\mstr@lang@meta]{contactcity}{%}
  \hypersetup{pdfcontactcity=
    \metapick[\mstr@lang@meta]{contactcity}}{}

- \metaifpick[\mstr@lang@meta]{contactregion}{%}
  \hypersetup{pdfcontactregion=
    \metapick[\mstr@lang@meta]{contactregion}}{}

- \metaifpick[\mstr@lang@meta]{contactcountry}{%}
  \hypersetup{pdfcontactcountry=
    \metapick[\mstr@lang@meta]{contactcountry}}{}

- \metaifpick[\mstr@lang@meta]{contactemail}{%}
  \hypersetup{pdfcontactemail=
    \metapick[\mstr@lang@meta]{contactemail}}{}

- \metaifpick[\mstr@lang@meta]{contacturl}{%}
  \hypersetup{pdfcontacturl=
    \metapick[\mstr@lang@meta]{contacturl}}%
B.7 Extras

The following defines some extras to be activated by package options.

**Course Metadata.** Include structures for course materials:

\begin{verbatim}
\ifmstr@opt@course
  \metadef{institution}
  \metadef{instructor}
  \metadef{course}
  \metadef{material}
  \metadef{period}
\end{verbatim}

Declare course structures:

\begin{verbatim}
\metadef{institution}
\metadef{instructor}
\metadef{course}
\metadef{material}
\metadef{period}
\end{verbatim}

Preset formatting styles:

\begin{verbatim}
\metaset{style}{course}{\LARGE\bfseries}
\metaset{style}{material}{\large}
\metaset{style}{institution}{\large}
\metaset{style}{period}{\large}
\metaset{skip}{material}{\vspace{2ex}}
\metaset{skip}{institution}{\vspace{4ex}}
\metaset{skip}{period}{\vspace{4ex}}
\metaset{skip}{instructor}{\vspace{6ex}}
\end{verbatim}

Fill `titletext` in `course` variant to display relevant title data for the course material:

\begin{verbatim}
\metaset{course}{titletext}{%
\metatitleline{print}{course}%
\metatitlelinetwo{print}{material}{draft}%
\metatitlelinetwo{print}{institution}{period}%
\metatitlelinetwo{print}{instructor}%
\end{verbatim}

Inherit title, subtitle, author and date:

\begin{verbatim}
\metaset{title}{\metapick[#1]{course}}
\metaset{subtitle}{\metapick[#1]{material}}
\metaset{author}{\metapick[#1]{instructor}}
\metaset{date}{\metapick[#1]{period}}
\metaset{location}{\metapick[#1]{institution}}
\end{verbatim}

B.8 Translations

Determine all desired international versions to be loaded; use first one as fallback language:

\begin{verbatim}
\end{verbatim}
English. Check whether to load English strings:

708 \ifdef\mstr@lang@en

Terms:

709 \metasetterm[en]{title}{Title}
710 \metasetterm[en]{abstract}{Abstract}
711 \metasetterm[en]{copyright}{Copyright}
712 \metasetterm[en]{preface}{Preface}
713 \metasetterm[en]{part}{Part}
714 \metasetterm[en]{chapter}{Chapter}
715 \metasetterm[en]{section}{Section}
716 \metasetterm[en]{subsection}{Subsection}
717 \metasetterm[en]{paragraph}{Paragraph}
718 \metasetterm[en]{appendix}{Appendix}
719 \metasetterm[en]{page}{Page}
720 \metasetterm[en]{figure}{Figure}
721 \metasetterm[en]{table}{Table}
722 \metasetterm[en]{contents}{Contents}
723 \metasetterm[en]{listfigure}{List of Figures}
724 \metasetterm[en]{listtable}{List of Tables}
725 \metasetterm[en]{references}{References}
726 \metasetterm[en]{index}{Index}
727 \metasetterm[en]{draft}{DRAFT}

General purpose messages:

728 \metaset[en]{urlmessage}\%
729 The current version of this work can be found at: \metapick[#1]{url}.
730 \metaset[en]{partofmessage}\%
731 This document is part of the work: \metapick[#1]{partof}.
732 \metaset[en]{licenseurlmessage}\%
733 To view a copy of this license, visit: \metapick[#1]{licenseurl}.

Copyright statements:

734 \metaset[en]{copyright@plain}\%
735 This work is protected by copyright.
736 \metaset[en]{copyright@parts}\%
737 This work as well as its parts is protected by copyright.
738 \metaset[en]{copyright@doc}\%
739 This document is protected by copyright.
740 \metaset[en]{copyright@doc-parts}\%
741 This document as well as its parts is protected by copyright.
742 \metaset[en]{copyright@reserved}{All rights reserved.}
743 \metaset[en]{copyright@publicdomain}{This work is dedicated to the public domain.}

License statements:

744 \metaset[en]{license@consent}\%
745 Reproduction of any part of this work in any form
746 without prior written consent
747 \metacompose[#1]{licenseprovider}{(of the author)
Reproduction of any part of this work in any form without prior written consent is not permissible.

This work may be distributed and/or modified under the conditions of the LaTeX Project Public License, either version \texttt{1.3} of this license or (at your option) any later version.

Creative Commons license composition:

\begin{verbatim}
\metaset[en]{license@consent-noncom}{}
\metacompose[#1]{licenseprovider}{}{}{of the author}
\metaset[en]{license@lppl}{}
\metaset[en]{license@pdpl}{}
\metaset[en]{license@noncom}{}
\fi
\end{verbatim}

**German.** Check whether to load German strings:

\begin{verbatim}
\ifdef\mstr\lang\de
Terms:
\end{verbatim}
French.  Disclaimer: professional assistance with translations needed.

Check whether to load French strings:

868 \ifdef\mstr@lang@fr

Terms:

869 \masetterm[fr]{title}{Titre}
870 \masetterm[fr]{abstract}{Résumené}
871 \masetterm[fr]{copyright}{Droits d'Auteur}
872 \masetterm[fr]{preface}{Préface}
873 \masetterm[fr]{part}{Partie}
874 \masetterm[fr]{chapter}{Chapitre}
875 \masetterm[fr]{section}{Section}
876 \masetterm[fr]{subsection}{Sous-Section}
877 \masetterm[fr]{paragraph}{Paragraphe}
878 \masetterm[fr]{appendix}{Annexe}
879 \masetterm[fr]{page}{Page}
880 \masetterm[fr]{figure}{Figure}
881 \masetterm[fr]{table}{Table}
882 \masetterm[fr]{contents}{Table des Matérieles}
883 \masetterm[fr]{listfigure}{Table des Figures}
884 \masetterm[fr]{listtable}{Liste des Tableaux}
885 \masetterm[fr]{references}{Références}
886 \masetterm[fr]{index}{Index}
887 \masetterm[fr]{draft}{BROUILLON}

General purpose messages:

888 \maset[fr]{urlmessage}{La version actuelle de cet œuvre se trouve à l'adresse:
889 \metapick[#1]{url}.}
890 \maset[fr]{partofmessage}{Ce document fait partie de la œuvre: \metapick[#1]{partof}.}
891 \maset[fr]{licenseurlmessage}{Pour voir une copie de cette licence, visitez:
892 \metapick[#1]{licenseurl}.}