The Org-article LaTeX class

Tom Dye

2010-09-18 Sat

Contents

1 Org-article class

This file uses Babel to tangle a LaTeX class named `org-article.cls`. The class provides all of the LaTeX packages used by the Org-mode LaTeX exporter. The class accepts numerous options, which the user can set either in `.emacs`,

for default use with all org-article exports, or within the Org-mode file using `#+LaTeX_CLASS_OPTIONS:`.

- Options to keep individual LaTeX packages from loading make it

somewhat easier to modify which LaTeX packages are loaded during the processing of LaTeX code exported by Org-mode.

- The package also implements several choices of fonts and takes

steps to ensure that the fonts don't clash with the symbol font files that Org-mode depends upon to typeset `org-entities`.

- The class provides pre-defined themes for formatting source code

listings, which can be used as is, or used as a bais for minor modifications.

- Facilities to typeset lists with less vertical space and to alter line spacing are also provided.

1.1 How to use this Org-mode document

You might be able to get the Org-mode document from GitHub using the following shell command, which works for me:

get-from-github() ≡

```
git clone git@github.com:tsdye/org-article.git
```

Alternatively, this might work for you:

get-from-github-alt() ≡

```
git clone git://github.com/tsdye/org-article.git
```

This will create a sub-directory, `org-article`, initialize the git repository and download the file `article-class.org` as part of the repository. The Org-mode document can be tangled to produce the `org-article.cls` file. This is done by running `org-babel-tangle` against the file, either by `M-x org-babel-tangle RET` or `C-c C-v [C-]t`.

The resulting `org-article.cls` file should then be moved where LaTeX can find it. In LaTeX setups that conform to the Tex Directory Structure, this might be `path/to/texmf-local/tex/latex/base`. Once the file has been placed in an appropriate directory it is often the case that the directory database, such

as the one maintained by Kpathsea, must then be updated. The following shell commands work on my OS-X system with the MacTeX distribution:

install-org-article() ≡

```
sudo cp org-article.cls /usr/local/texlive/texmf-local/tex/latex
   /base/
sudo mktexlsr
kpsewhich org-article.cls
```

1.2 Org-mode LaTeX export setup

There are two ways to setup `org-article.cls` and your choice will probably depend on the value of the variable `org-export-latex-packages-alist`. If this variable is nil (or it refers to packages that you always want loaded), then the following setup should work for you. It asks Org-mode **not** to load the default packages, because these are loaded by `org-article.cls`. Then, it loads the packages in `org-export-latex-packages-alist`, which should consist of a single entry for the `inputenc` package. Org-mode automatically sets the input encoding based on the status of the Org-mode buffer being exported, which it can't do if it is loaded by `org-article.cls`. Finally, any packages specified in the Org-mode buffer are loaded (the [EXTRA] argument).

The `org-article.cls` setup for the case when `org-export-latex-packages-alist` is `nil`:

```
(add-to-list 'org-export-latex-packages-alist
             '(("AUTO" "inputenc" t)))
(add-to-list 'org-export-latex-classes
          '("org-article-section"
             "\\documentclass{org-article}
             \\loadpackage[AUTO]{inputenc}
             [NO-DEFAULT-PACKAGES]
             [PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-subsection"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")))
```

```
(add-to-list 'org-export-latex-classes
          '("org-article-subsubsection"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")
             ("\\subsubsection{%s}" . "\\subsubsection*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-paragraph"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")
             ("\\subsubsection{%s}" . "\\subsubsection*{%s}")
             ("\\paragraph{%s}" . "\\paragraph*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-subparagraph"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")
             ("\\subsubsection{%s}" . "\\subsubsection*{%s}")
             ("\\paragraph{%s}" . "\\paragraph*{%s}")
             ("\\subparagraph{%s}" . "\\subparagraph*{%s}")))
```

If, for some reason, `org-export-latex-packages-alist` is not `nil`, and it includes packages that you don't want always loaded then the following setup should work.

```
(add-to-list 'org-export-latex-classes
          '("org-article-section"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-subsection"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [EXTRA]"
```

```
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-subsubsection"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")
             ("\\subsubsection{%s}" . "\\subsubsection*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-paragraph"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")
             ("\\subsubsection{%s}" . "\\subsubsection*{%s}")
             ("\\paragraph{%s}" . "\\paragraph*{%s}")))
(add-to-list 'org-export-latex-classes
          '("org-article-subparagraph"
             "\\documentclass{org-article}
             [NO-DEFAULT-PACKAGES]
             [EXTRA]"
             ("\\section{%s}" . "\\section*{%s}")
             ("\\subsection{%s}" . "\\subsection*{%s}")
             ("\\subsubsection{%s}" . "\\subsubsection*{%s}")
             ("\\paragraph{%s}" . "\\paragraph*{%s}")
             ("\\subparagraph{%s}" . "\\subparagraph*{%s}")))
```

In this case, you will need to specify the `inputenc` package in the Org-mode file:

specify-inputenc() ≡

```
#+LATEX_HEADER: \usepackage[utf8]{inputenc}
```

1.3 Examples

The functionality of `org-article.cls` can be demonstrated with the following two examples of pdf output generated by Org-mode LaTeX export of this Org-mode file. In the first, these two lines were included at the top of the Org-mode file:

first-example() ≡

```
#+LaTeX_CLASS: org-article-subsubsection
```

```
#+LaTeX_CLASS_OPTIONS: [article,letterpaper,times,12pt,listings-
    bw,microtype]
```

The resulting pdf file is typeset with the standard LaTeX `article.cls` on 8.5 x 11 in. paper, using Times, Helvetica, and Courier fonts with a 12 point base size. Source code listings are given in black and white, and microtypographic justification is applied.

In the second example, the following two lines were included in the top of the Org-mode file:

second-example() ≡

```
#+LaTeX_CLASS: org-article-subsubsection
#+LaTeX_CLASS_OPTIONS: [koma,a4paper,landscape,twocolumn,utopia
    ,10pt,listings-sv,microtype,paralist]
```

The resulting pdf file is typeset with the KOMA-script `scrartcl.cls` on 5.8 x 8.3 in. paper in landscape mode, using Utopia, Bera, and Incosolata fonts with a 10 point base size. Source code listings are given in color, and microtypographic justification is applied. In addition, the `paralist` option has been set; compare the tightly-set list immediately below with the standard list of the first example.

1.4 The class file

The LaTeX class file has six standard parts:

Identification part Defines the nature of the file and specifies the TeX format that it requires.

Initial code part Loads packages used internally by the class file.

Declaration of options part All options known to the class are declared here. It is forbidden to load packages in this part.

Execution of options part Set default values and execute the code for the options that have been declared.

Package loading part Load packages with the options specified in the declaration of options part using `\PassOptionsToPackage`.

Main code part Usually used to define new commands and structures.

1.4.1 Identification part

This is a standard identification part. The `\NeedsTeXFormat` command can take an optional argument with a release date for the oldest version of LaTeX that can use the class. Since it is relatively easy to update LaTeX installations nowadays there is less reason to use this optional argument than there was in the past. It is omitted here.

identification-part() ≡

```
% Identification part
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{org-article}[2010/09/18 0.2 (TSD)]
% End of the identification part
%
```

1.4.2 Initial code part

The initial code part loads packages needed to process the class file and declares booleans for each of the class options. Options defined by the base class, either the standard `article.cls` or the Koma class `srcartcl.cls`, are passed on to those classes by default and don't have to be declared here.

initial-code-part() ≡

```
% Initial code part

\RequirePackage{ifthen}
\RequirePackage{calc}
\RequirePackage{ifpdf}

% End of initial code part
```

1.4.3 Declaration of options part

The package options are declared here in a code block made up entirely of noweb references. Typically, a package referred to here will also appear in the package loading part. The package loading part also consists of noweb references, an arrangement that makes it possible to keep all the code specific to a particular package together in the LaTeX packages section.

1.4.4 Execution of options part

The `\ProcessOptions` command reclaims the memory used to store user options, so those values are now gone unless something was done with them in the declaration of options part.

execution-of-options-part() ≡

```
% Execution of options part

\ProcessOptions\relax


% End of execution of options part
```

1.4.5 Package loading part

By default, `org-article.cls` loads all but one of the packages in `org-export-latex-default-`
It does not load `inputenc` directly, but instead relies on the Org-mode LaTeX exporter to load this package, which passes as an option the encoding scheme of the exported buffer. The `fontenc` package is loaded with the T1 option by default as a prerequisite for the various symbol packages. There is no facility to disable loading `fontenc`, which is unusual among LaTeX packages in its ability to be loaded more than once. This functionality is required in the case where two or more fonts with different encodings are used.

This code block is implemented as noweb references so that package-specific code can be kept together in LaTeX packages.

1.4.6 Class code part

This part is also implemented with noweb references. It calls package-specific setup routines that are defined in the LaTeX packages section.

1.5 LaTeX packages

1.5.1 Article base class options

`Org-article.cls` offers a choice of two base classes. The first is the standard LaTeX `article.cls`. Also available is the KOMA-script `scrartcl.cls`. The KOMA-script `scrartcl.cls` is compatible with the standard LaTeX article class; input that compiles with `article.cls` should also compile with

`scrartcl.cls`. It differs in the layout of the page and the styling of page elements, producing a somewhat more "modern" design based on principles set out by the typographer and book designer Jan Tschichold.

To select the standard LaTeX `article.cls`, put this in your Org-mode document:

org-buffer-article() ≡

```
#+LaTeX_CLASS_OPTIONS: [article]
```

To select the KOMA-script `scrartcl.cls`, put this in your Org-mode document:

org-buffer-koma() ≡

```
#+LaTeX_CLASS_OPTIONS: [koma]
```

For information on the KOMA-script `scrartcl.cls`, you can probably read the documentation on your system with the following shell command:

read-koma() ≡

```
texdoc koma
```

option-koma() ≡

```
\newboolean{koma}
\DeclareOption{koma}{\setboolean{koma}{true}}
```

option-article() ≡

```
\newboolean{article}
\DeclareOption{article}{\setboolean{article}{true}}
```

pass-to-koma() ≡

```
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{scrartcl}}
```

pass-to-article() ≡

```
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
```

The article class is loaded by default.

load-base-class() ≡

```
\ifthenelse{\boolean{koma}}
{%
  \LoadClass{scrartcl}%
}%
{%
```

```
\LoadClass{article}%
}
```

- Paper size
  The following paper size options are available for the standard LaTeX `article.cls` and the KOMA-script `scrartcl.cls`. The first three options are North American paper sizes. The `a4paper`, `a5paper`, `b4paper`, and `b5paper` options are international standard ISO 216. The `landscape` option orients the paper with the long axis horizontal.

  paper-sizes() ≡

```
#+LaTeX_CLASS_OPTIONS: [letterpaper]
#+LaTeX_CLASS_OPTIONS: [legalpaper]
#+LaTeX_CLASS_OPTIONS: [executivepaper]
#+LaTeX_CLASS_OPTIONS: [a4paper]
#+LaTeX_CLASS_OPTIONS: [a5paper]
#+LaTeX_CLASS_OPTIONS: [b4paper]
#+LaTeX_CLASS_OPTIONS: [b5paper]
#+LaTeX_CLASS_OPTIONS: [landscape]
```

  The KOMA-script `scrartcl.cls` has options for a fuller range of the international standard ISO 216 paper sizes, in addition to the `a4paper`, `a5paper`, `b4paper`, and `b5paper` sizes offered by the standard LaTeX `article.cls`. In the example below, X is replaced by an integer [0, 1, ... 10].

  koma-paper-sizes() ≡

```
#+LaTeX_CLASS_OPTIONS: [aXpaper]
#+LaTeX_CLASS_OPTIONS: [bXpaper]
#+LaTeX_CLASS_OPTIONS: [cXpaper]
#+LaTeX_CLASS_OPTIONS: [dXpaper]
```

- Font size
  There are three base font size options available for the standard LaTeX `article.cls` and the KOMA-script `scrartcl.cls`. This option sets the size of the main text in the body of the document. Other fonts used in the document design, such as headers, footers, heads, sub-heads, etc., will be scaled accordingly.

  font-sizes() ≡

```
#+LaTeX_CLASS_OPTIONS: [10pt]
#+LaTeX_CLASS_OPTIONS: [11pt]
#+LaTeX_CLASS_OPTIONS: [12pt]
```

- Equations
  The standard LaTeX `article.cls` and the KOMA-script `scrartcl.cls` both recognize two options that control formatting of equations. The option `leqno` will number equations on the left, rather than the right, which is the default. The option `fleqn` displays equations flush left, rather than centered, which is the default

  equations() ≡

```
#+LaTeX_CLASS_OPTIONS: [leqno]
#+LaTeX_CLASS_OPTIONS: [fleqn]
```

- Table captions
  The standard LaTeX `article.cls` formats captions to appear below the captioned item. However, many document styles require table captions above the table. Users of `article.cls` typically use a package, =topcapt.sty=, and place the command `\topcaption{}` above the captioned item. With the Org-mode LaTeX exporter, this requires changes to the exported LaTeX code. The KOMA-script `scrartcl.cls` provides an option that gets rid of the need for `topcapt.sty`, but the code produced by the LaTeX exporter must still be changed to place the caption above the table within the `table` environment:

  koma-caption() ≡

```
#+LaTeX_CLASS_OPTIONS: [captions=tableheading]
```

1.5.2 Org-mode default packages

- Inputenc
  The input encoding of the document is specified by the `inputenc` package. Org-mode provides a nifty method for sending options to this package, so it is not loaded directly by `org-article.cls`. See Org-mode LaTeX export setup.

- Fontenc
  The `fontenc` package specifies the encoding to use with a font. The history of font encodings in LaTeX is a long one; suffice it to say that the most common option is `T1`, also known as the Cork encoding because it was formulated at a EuroTeX conference in Ireland's County Cork. The `fontenc` package pretends that it was never loaded so that it can be called several times with different options to load fonts that have various encodings.

  You can probably read the documentation for `fontenc` on your system with the following shell command:

  read-fontenc() ≡

```
texdoc fontenc
```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-fontenc() ≡

```
#+LaTeX_CLASS_OPTIONS: [nofontenc]
```

  Note that several of the font packages load `fontenc` themselves. These include Garamond, Palatino, Charter, and Utopia.

  option-fontenc() ≡

```
\newboolean{nofontenc}
\DeclareOption{nofontenc}{\setboolean{nofontenc}{true}}
```

  load-fontenc() ≡

```
\ifthenelse{\boolean{nofontenc}}
{}
{\RequirePackage[T1]{fontenc}}
```

  options-to-fontenc() ≡

```
\DeclareOption*{%
  \PassOptionsToPackage{\CurrentOption}{fontenc}
}
```

- Fixltx2e
  The `fixltx2e` package applies fixes to LaTeX2e that would break older

documents, so have not been applied to the LaTeX2e kernel. The package doesn't take any options.

You can probably read about `fixltx2e` on your system by issuing the following shell command:

read-fixltx2e() ≡

```
texdoc fixltx2e
```

This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

org-buffer-fixltx2e() ≡

```
#+LaTeX_CLASS_OPTIONS: [nofixltx2e]
```

option-fixltx2e() ≡

```
\newboolean{nofixltx2e}
\DeclareOption{nofixltx2e}{\setboolean{nofixltx2e}{true}}
```

load-fixltx2e() ≡

```
\ifthenelse{\boolean{nofixltx2e}}
{}
{\RequirePackage{fixltx2e}}
```

- Graphicx
  The `graphicx` package is typically configured with *.def files because the facilities it specifies are provided by a graphics driver, rather than by LaTeX. For this reason, it is typically loaded without options.

  You should be able to read about `graphicx`, along with its companion packages `color` and `graphics` by issuing the following shell command:

  read-graphicx() ≡

  ```
  texdoc graphicx
  ```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-graphicx() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [nographicx]
  ```

option-graphicx() ≡

```
\newboolean{nographicx}
\DeclareOption{nographicx}{\setboolean{nographicx}{true}}
```

load-graphicx() ≡

```
\ifthenelse{\boolean{nographicx}}
{}
{\RequirePackage{graphicx}}
```

- Longtable
  The `longtable` package defines a new LaTeX environment that can be used in place of the `tabular` environment and can be broken by the TeX page-breaking algorithm. It is used, as the name implies, by long tables that typically won't fit onto a single page. The package is loaded without option.

  You should be able to read the `longtable` documentation on your system by issuing the following shell command:

  read-longtable() ≡

  ```
  texdoc longtable
  ```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-longtable() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [nolongtable]
  ```

  option-longtable() ≡

  ```
  \newboolean{nolongtable}
  \DeclareOption{nolongtable}{\setboolean{nolongtable}{true}}
  ```

  load-longtable() ≡

  ```
  \ifthenelse{\boolean{nolongtable}}
  {}
  {\RequirePackage{longtable}}
  ```

- Float
  Tables and figures in LaTeX are treated as floating objects. Internally, they

are treated as a single (large) glyph, which makes them difficult to place on a page of otherwise small glyphs. Consequently, they are allowed to "float" until a suitable location is found. The `float` package provides facilities to define new floating environments, to restyle the existing float environments, and additionally defines a placement parameter, `[H]`, that keeps a float from floating. The package is loaded without options.

You can probably read about the `float` package on your system by issuing the following shell command:

read-float() ≡

```
texdoc float
```

This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

org-buffer-float() ≡

```
#+LaTeX_CLASS_OPTIONS: [nofloat]
```

option-float() ≡

```
\newboolean{nofloat}
\DeclareOption{nofloat}{\setboolean{nofloat}{true}}
```

load-float() ≡

```
\ifthenelse{\boolean{nofloat}}
{}
{\RequirePackage{float}}
```

- Wrapfig
  The `wrapfig` package defines two new environments to set a narrow float at the edge of the text and wrap the text around it. Because "floats" in these new environments do not float it is sometimes the case that they appear out of order, e.g. `Figure n` appears before `Figure n-1`. Caveat emptor.

  The package is loaded without options.

  The documentation for this package is included at the end of the package source. You should be able to read it on your system by issuing the following shell command:

  read-wrapfig() ≡

```
texdoc wrapfig
```

This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

org-buffer-wrapfig() ≡

```
#+LaTeX_CLASS_OPTIONS: [nowrapfig]
```

option-wrapfig() ≡

```
\newboolean{nowrapfig}
\DeclareOption{nowrapfig}{\setboolean{nowrapfig}{true}}
```

load-wrapfig() ≡

```
\ifthenelse{\boolean{nowrapfig}}
{}
{\RequirePackage{wrapfig}}
```

- Soul
  The `soul` package is used primarily for underlining text. It is loaded without options.

  You can probably read the `soul` documentation on your system by issuing the following shell command:

  read-soul() ≡

  ```
  texdoc soul
  ```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-soul() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [nosoul]
  ```

  option-soul() ≡

  ```
  \newboolean{nosoul}
  \DeclareOption{nosoul}{\setboolean{nosoul}{true}}
  ```

  load-soul() ≡

```
\ifthenelse{\boolean{nosoul}}
{}
{\RequirePackage{soul}}
```

- Textcomp
  This package provides support for the Text Companion fonts, which provide symbols used by `org-entities`, in particular the Euro currency symbol. It is loaded without options.

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-textcomp() ≡

```
#+LaTeX_CLASS_OPTIONS: [notextcomp]
```

  option-textcomp() ≡

```
\newboolean{notextcomp}
\DeclareOption{notextcomp}{\setboolean{notextcomp}{true}}
```

  load-textcomp() ≡

```
\ifthenelse{\boolean{notextcomp}}
{}
{\RequirePackage{textcomp}}
```

- MarVoSym
  The `marvosym` package provides support for Martin Vogel's Symbol font, some glyphs from which are required by `org-entities`. The package is loaded without options.

  You can probably read about the `marvosym` package by issuing the following command in the shell:

  read-marvosym() ≡

```
texdoc marvosym
```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-marvosym() ≡

```
#+LaTeX_CLASS_OPTIONS: [nomarvosym]
```

option-marvosym() ≡

```
\newboolean{nomarvosym}
\DeclareOption{nomarvosym}{\setboolean{nomarvosym}{true}}
```

load-marvosym() ≡

```
\ifthenelse{\boolean{nomarvosym}}
{}
{\RequirePackage{marvosym}}
```

- Wasysym
  The `wasysym` package makes available some symbol glyphs from the `wasy` fonts. It is needed to support some of the glyphs in `org-entities`. When it is loaded without options, this package clashes with the American Mathematical Society's `amsmath` package. Using the `nointegrals` option resolves this clash:

  wasysym-options() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [integrals, nointegrals]
  ```

  You can probably read the wasysym documentation on your system by issuing the following shell command:

  read-wasysym() ≡

  ```
  texdoc wasysym
  ```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-wasysym() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [nowasysym]
  ```

  option-wasysym() ≡

  ```
  \newboolean{nowasysym}
  \DeclareOption{nowasysym}{\setboolean{nowasysym}{true}}
  ```

  load-wasysym() ≡

```
\ifthenelse{\boolean{nowasysym}}
{}
{\RequirePackage[nointegrals]{wasysym}}
```

- Latexsym
  The `latexsym` package provides a few glyphs, one or more of which might be required by `org-entities`. According to the documentation, `latexsym` isn't needed if the `amssymb` package is loaded.

  You can probably read about the `latexsym` package on your system by issuing the following shell command:

  read-latexsym() ≡

```
texdoc latexsym
```

  This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

  org-buffer-latexsym() ≡

```
#+LaTeX_CLASS_OPTIONS: [nolatexsym]
```

  option-latexsym() ≡

```
\newboolean{nolatexsym}
\DeclareOption{nolatexsym}{\setboolean{nolatexsym}{true}}
```

  load-latexsym() ≡

```
\ifthenelse{\boolean{nolatexsym}}
{}
{\RequirePackage{latexsym}}
```

- Amssymb
  This package provides all the symbols defined in the American Mathematical Society's symbol fonts `msam` and `msbm`. They are required to support `org-entities`. It is superseded by the `mathdesign` package, which is used by various fonts. If one of these is specified, then the `amssymb` package is not loaded. If the package is loaded, the it is loaded without options.

You can probably read the amssymb package documentation by issuing the following shell command:

read-amssymb() ≡

```
texdoc amssymb
```

This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

org-buffer-amssymb() ≡

```
#+LaTeX_CLASS_OPTIONS: [noamssymb]
```

option-amssymb() ≡

```
\newboolean{noamssymb}
\DeclareOption{noamssymb}{\setboolean{noamssymb}{true}}
```

Isn't loaded if Times, Charter, Utopia, or Garamond are loaded. These use the mathdesign package, which apparently supersedes amssymb.

load-amssymb() ≡

```
\ifthenelse{\boolean{noamssymb}\or\boolean{utopia}\or\
   boolean{charter}\or\boolean{garamond}\or\boolean{times}}
{}
{\RequirePackage{amssymb}}
```

- Hyperref
  The hyperref package turns LaTeX cross-referencing commands into hyperlinks, including the table of contents, bibliography, etc. It is typically configured on a site-wide basis with options kept in a file, hyperref.cfg. The LaTeX document loads the package without specifying any options. The hyperref package redefines many LaTeX commands, so it needs to be loaded at, or near the end of, the package loading part.

  The hyperref package accepts numerous options, which can be given as key = value pairs. Boolean options default to true when passed without a value. Options are passed in the usual way, and org-article.cls simply passes them on to hyperref.

  hyperref-options() ≡

```
#+LaTeX_CLASS_OPTIONS: [anchorcolor, backref, baseurl,
   bookmarks,
bookmarksnumbered, bookmarksopen, bookmarksopenlevel,
   bookmarkstype,
breaklinks, CJKbookmarks, citebordercolor, citecolor,
   colorlinks,
draft, dvipdfm, dvipdfmx, dvips, dvipsone, dviwindo, encap,
extension, filebordercolor, filecolor, final, frenchlinks,
hyperfigures, hyperfootnotes, hyperindex, hypertex,
   hypertexnames,
implicit, latex2html, legalpaper, letterpaper,
   linkbordercolor,
linkcolor, linktocpage, menubordercolor, menucolor,
   nativepdf,
naturalnames, nesting, pageanchor, pagebackref, pdfauthor,
pdfborder, pdfcenterwindow, pdfcreator, pdfdirection,
pdfdisplaydoctitle, pdfduplex, pdffitwindow, pdfhighlight,
   pdfinfo,
pdfkeywords, pdflang, pdfmark, pdfmenubar, pdfnewwindow,
pdfnonfullscreenpagemode, pdfnumcopies, pdfpagelayout,
   pdfpagemode,
pdfpagelabels, pdfpagescrop, pdfpagetransition,
pdfpicktraybypdfsize, pdfprintarea, pdfprintclip,
   pdfprintpagerange,
pdfprintscaling, pdfproducer, pdfstartpage, pdfstartview,
pdfsubject, pdftex, pdftitle, pdftoolbar, pdftrapped,
   pdfview,
pdfviewarea, pdfviewclip, pdfwindowui, plainpages, ps2pdf,
raiselinks, runbordercolor, runcolor, setpagesize, tex4ht,
   textures,
unicode, urlbordercolor, urlcolor, verbose, vtex, xetex]
```

You can probably read the `hyperref` documentation by issuing the following shell command:

read-hyperref() ≡

```
texdoc hyperref
```

This is a standard Org-mode package that is loaded by default. An option is provided to not load it.

org-buffer-hyperref() ≡

```
#+LaTeX_CLASS_OPTIONS: [nohyperref]
```

option-hyperref() ≡

```
\newboolean{nohyperref}
\DeclareOption{nohyperref}{\setboolean{nohyperref}{true}}
```

load-hyperref() ≡

```
\ifthenelse{\boolean{nohyperref}}
{}
{\RequirePackage{hyperref}}
```

Options do not include `debug`.

options-to-hyperref() ≡

```
\DeclareOption{anchorcolor}{%
   \PassOptionsToPackage{anchorcolor}{hyperref}}
\DeclareOption{backref}{%
   \PassOptionsToPackage{backref}{hyperref}}
\DeclareOption{baseurl}{%
   \PassOptionsToPackage{baseurl}{hyperref}}
\DeclareOption{bookmarks}{%
   \PassOptionsToPackage{bookmarks}{hyperref}}
\DeclareOption{bookmarksnumbered}{%
   \PassOptionsToPackage{bookmarksnumbered}{hyperref}}
\DeclareOption{bookmarksopen}{%
   \PassOptionsToPackage{bookmarksopen}{hyperref}}
\DeclareOption{bookmarksopenlevel}{%
   \PassOptionsToPackage{bookmarksopenlevel}{hyperref}}
\DeclareOption{bookmarkstype}{%
   \PassOptionsToPackage{bookmarkstype}{hyperref}}
\DeclareOption{breaklinks}{%
   \PassOptionsToPackage{breaklinks}{hyperref}}
\DeclareOption{CJKbookmarks}{%
   \PassOptionsToPackage{CJKbookmarks}{hyperref}}
\DeclareOption{citebordercolor}{%
   \PassOptionsToPackage{citebordercolor}{hyperref}}
\DeclareOption{citecolor}{%
   \PassOptionsToPackage{citecolor}{hyperref}}
\DeclareOption{colorlinks}{%
   \PassOptionsToPackage{colorlinks}{hyperref}}
\DeclareOption{draft}{%
   \PassOptionsToPackage{draft}{hyperref}}
\DeclareOption{dvipdfm}{%
   \PassOptionsToPackage{dvipdfm}{hyperref}}
\DeclareOption{dvipdfmx}{%
```

```
   \PassOptionsToPackage{dvipdfmx}{hyperref}}
\DeclareOption{dvips}{%
   \PassOptionsToPackage{dvips}{hyperref}}
\DeclareOption{dvipsone}{%
   \PassOptionsToPackage{dvipsone}{hyperref}}
\DeclareOption{dviwindo}{%
   \PassOptionsToPackage{dviwindo}{hyperref}}
\DeclareOption{encap}{%
   \PassOptionsToPackage{encap}{hyperref}}
\DeclareOption{extension}{%
   \PassOptionsToPackage{extension}{hyperref}}
\DeclareOption{filebordercolor}{%
   \PassOptionsToPackage{filebordercolor}{hyperref}}
\DeclareOption{filecolor}{%
   \PassOptionsToPackage{filecolor}{hyperref}}
\DeclareOption{final}{%
   \PassOptionsToPackage{final}{hyperref}}
\DeclareOption{frenchlinks}{%
   \PassOptionsToPackage{frenchlinks}{hyperref}}
\DeclareOption{hyperfigures}{%
   \PassOptionsToPackage{hyperfigures}{hyperref}}
\DeclareOption{hyperfootnotes}{%
   \PassOptionsToPackage{hyperfootnotes}{hyperref}}
\DeclareOption{hyperindex}{%
   \PassOptionsToPackage{hyperindex}{hyperref}}
\DeclareOption{hypertex}{%
   \PassOptionsToPackage{hypertex}{hyperref}}
\DeclareOption{hypertexnames}{%
   \PassOptionsToPackage{hypertexnames}{hyperref}}
\DeclareOption{implicit}{%
   \PassOptionsToPackage{implicit}{hyperref}}
\DeclareOption{latex2html}{%
   \PassOptionsToPackage{latex2html}{hyperref}}
\DeclareOption{legalpaper}{%
   \PassOptionsToPackage{legalpaper}{hyperref}}
\DeclareOption{letterpaper}{%
   \PassOptionsToPackage{letterpaper}{hyperref}}
\DeclareOption{linkbordercolor}{%
   \PassOptionsToPackage{linkbordercolor}{hyperref}}
\DeclareOption{linkcolor}{%
   \PassOptionsToPackage{linkcolor}{hyperref}}
\DeclareOption{linktocpage}{%
   \PassOptionsToPackage{linktocpage}{hyperref}}
\DeclareOption{menubordercolor}{%
   \PassOptionsToPackage{menubordercolor}{hyperref}}
```

```
\DeclareOption{menucolor}{%
   \PassOptionsToPackage{menucolor}{hyperref}}
\DeclareOption{nativepdf}{%
   \PassOptionsToPackage{nativepdf}{hyperref}}
\DeclareOption{naturalnames}{%
   \PassOptionsToPackage{naturalnames}{hyperref}}
\DeclareOption{nesting}{%
   \PassOptionsToPackage{nesting}{hyperref}}
\DeclareOption{pageanchor}{%
   \PassOptionsToPackage{pageanchor}{hyperref}}
\DeclareOption{pagebackref}{%
   \PassOptionsToPackage{pagebackref}{hyperref}}
\DeclareOption{pdfauthor}{%
   \PassOptionsToPackage{pdfauthor}{hyperref}}
\DeclareOption{pdfborder}{%
   \PassOptionsToPackage{pdfborder}{hyperref}}
\DeclareOption{pdfcenterwindow}{%
   \PassOptionsToPackage{pdfcenterwindow}{hyperref}}
\DeclareOption{pdfcreator}{%
   \PassOptionsToPackage{pdfcreator}{hyperref}}
\DeclareOption{pdfdirection}{%
   \PassOptionsToPackage{pdfdirection}{hyperref}}
\DeclareOption{pdfdisplaydoctitle}{%
   \PassOptionsToPackage{pdfdisplaydoctitle}{hyperref}}
\DeclareOption{pdfduplex}{%
   \PassOptionsToPackage{pdfduplex}{hyperref}}
\DeclareOption{pdffitwindow}{%
   \PassOptionsToPackage{pdffitwindow}{hyperref}}
\DeclareOption{pdfhighlight}{%
   \PassOptionsToPackage{pdfhighlight}{hyperref}}
\DeclareOption{pdfinfo}{%
   \PassOptionsToPackage{pdfinfo}{hyperref}}
\DeclareOption{pdfkeywords}{%
   \PassOptionsToPackage{pdfkeywords}{hyperref}}
\DeclareOption{pdflang}{%
   \PassOptionsToPackage{pdflang}{hyperref}}
\DeclareOption{pdfmark}{%
   \PassOptionsToPackage{pdfmark}{hyperref}}
\DeclareOption{pdfmenubar}{%
   \PassOptionsToPackage{pdfmenubar}{hyperref}}
\DeclareOption{pdfnewwindow}{%
   \PassOptionsToPackage{pdfnewwindow}{hyperref}}
\DeclareOption{pdfnonfullscreenpagemode}{%
   \PassOptionsToPackage{pdfnonfullscreenpagemode}{hyperref
      }}
```

```
\DeclareOption{pdfnumcopies}{%
   \PassOptionsToPackage{pdfnumcopies}{hyperref}}
\DeclareOption{pdfpagelayout}{%
   \PassOptionsToPackage{pdfpagelayout}{hyperref}}
\DeclareOption{pdfpagemode}{%
   \PassOptionsToPackage{pdfpagemode}{hyperref}}
\DeclareOption{pdfpagelabels}{%
   \PassOptionsToPackage{pdfpagelabels}{hyperref}}
\DeclareOption{pdfpagescrop}{%
   \PassOptionsToPackage{pdfpagescrop}{hyperref}}
\DeclareOption{pdfpagetransition}{%
   \PassOptionsToPackage{pdfpagetransition}{hyperref}}
\DeclareOption{pdfpicktraybypdfsize}{%
   \PassOptionsToPackage{pdfpicktraybypdfsize}{hyperref}}
\DeclareOption{pdfprintarea}{%
   \PassOptionsToPackage{pdfprintarea}{hyperref}}
\DeclareOption{pdfprintclip}{%
   \PassOptionsToPackage{pdfprintclip}{hyperref}}
\DeclareOption{pdfprintpagerange}{%
   \PassOptionsToPackage{pdfprintpagerange}{hyperref}}
\DeclareOption{pdfprintscaling}{%
   \PassOptionsToPackage{pdfprintscaling}{hyperref}}
\DeclareOption{pdfproducer}{%
   \PassOptionsToPackage{pdfproducer}{hyperref}}
\DeclareOption{pdfstartpage}{%
   \PassOptionsToPackage{pdfstartview}{hyperref}}
\DeclareOption{pdfsubject}{%
   \PassOptionsToPackage{pdfsubject}{hyperref}}
\DeclareOption{pdftex}{%
   \PassOptionsToPackage{pdftex}{hyperref}}
\DeclareOption{pdftitle}{%
   \PassOptionsToPackage{pdftitle}{hyperref}}
\DeclareOption{pdftoolbar}{%
   \PassOptionsToPackage{pdftoolbar}{hyperref}}
\DeclareOption{pdftrapped}{%
   \PassOptionsToPackage{pdftrapped}{hyperref}}
\DeclareOption{pdfview}{%
   \PassOptionsToPackage{pdfview}{hyperref}}
\DeclareOption{pdfviewarea}{%
   \PassOptionsToPackage{pdfviewarea}{hyperref}}
\DeclareOption{pdfviewclip}{%
   \PassOptionsToPackage{pdfviewclip}{hyperref}}
\DeclareOption{pdfwindowui}{%
   \PassOptionsToPackage{pdfwindowui}{hyperref}}
\DeclareOption{plainpages}{%
```

```
   \PassOptionsToPackage{plainpages}{hyperref}}
\DeclareOption{ps2pdf}{%
   \PassOptionsToPackage{ps2pdf}{hyperref}}
\DeclareOption{raiselinks}{%
   \PassOptionsToPackage{raiselinks}{hyperref}}
\DeclareOption{runbordercolor}{%
   \PassOptionsToPackage{runbordercolor}{hyperref}}
\DeclareOption{runcolor}{%
   \PassOptionsToPackage{runcolor}{hyperref}}
\DeclareOption{setpagesize}{%
   \PassOptionsToPackage{setpagesize}{hyperref}}
\DeclareOption{tex4ht}{%
   \PassOptionsToPackage{tex4ht}{hyperref}}
\DeclareOption{textures}{%
   \PassOptionsToPackage{textures}{hyperref}}
\DeclareOption{unicode}{%
   \PassOptionsToPackage{unicode}{hyperref}}
\DeclareOption{urlbordercolor}{%
   \PassOptionsToPackage{urlbordercolor}{hyperref}}
\DeclareOption{urlcolor}{%
   \PassOptionsToPackage{urlcolor}{hyperref}}
\DeclareOption{verbose}{%
   \PassOptionsToPackage{verbose}{hyperref}}
\DeclareOption{vtex}{%
   \PassOptionsToPackage{vtex}{hyperref}}
\DeclareOption{xetex}{%
   \PassOptionsToPackage{xetex}{hyperref}}
```

1.5.3 Font packages

LaTeX documents might need three text fonts, one for the serif typeface used for text, the sans-serif typeface often used for heads and sub-heads, and the monospace typewriter typeface typically used to set code examples and the like. Each of the following options specifies all three of the fonts, but takes its name after the serif font used to set text.

- Times
  The `times` option uses URW Nimbus Roman, a Times clone, for the serif font, URW Nimbus Sans, a Helvetica clone, for the sans-serif font, and URW Nimbus Mono, a Courier clone, for the typewriter font. This is a standard set of common typefaces typically used in scientific publications. All of the fonts should be included in a typical LaTeX distribution.

Times New Roman was designed by Stanley Morison for *The Times* of London during a redesign of the newspaper prompted, in part, by Morison's criticism of its typography in 1929. Helvetica was developed in 1957 by Max Miedinger. Courier was designed by Howard Kettler in 1955 for use in IBM typewriters.

org-buffer-times() ≡

```
#+LaTeX_CLASS_OPTIONS: [times]
```

option-times() ≡

```
\newboolean{times}
\DeclareOption{times}{\setboolean{times}{true}}
```

Helvetica looks better if it is set slightly smaller than the serif font.

load-times() ≡

```
\ifthenelse{\boolean{times}}
{%
  \ifpdf
  \RequirePackage[T1]{fontenc}
  \RequirePackage{mathptmx}
  \RequirePackage[scaled=.90]{helvet}
  \RequirePackage{courier}
  \fi}%
{}
```

- Garamond
  Garamond refers to a group of old-style serif typefaces and is named after the sixteenth-century type designer, Claude Garamond. It is an elegant typeface. The sans-serif font is Bera, an adaptation of a font originally named Vera. It was designed by Jim Lyles. The typewriter font is Inconsolata, which was created by Raph Levien and is based on Vera.

  org-buffer-garamond() ≡

```
#+LaTeX_CLASS_OPTIONS: [garamond]
```

  option-garamond() ≡

```
\newboolean{garamond}
\DeclareOption{garamond}{\setboolean{garamond}{true}}
```

Garamond requires a bit more leading than normal.

load-garamond() ≡

```
\ifthenelse{\boolean{garamond}}
{%
  \ifpdf
   \RequirePackage[T1]{fontenc}
   \RequirePackage[urw-garamond]{mathdesign}
   \RequirePackage[scaled]{berasans}
   \RequirePackage{inconsolata} % tt
   \linespread{1.0609}
  \fi}%
{}
```

- Palatino
  The beautiful, old-style serif font, Palatino, was designed by Herman Zapf. It is somewhat heavier and easier to read than Garamond. It is paired here with Helvetica and Courier, as is Times, for which it is an alternative.

  org-buffer-palatino() ≡

```
#+LaTeX_CLASS_OPTIONS: [palatino]
```

  option-palatino() ≡

```
\newboolean{palatino}
\DeclareOption{palatino}{\setboolean{palatino}{true}}
```

  Palatino gets a bit more leading than normal.

  load-palatino() ≡

```
\ifthenelse{\boolean{palatino}}
{%
  \ifpdf
  \RequirePackage[T1]{fontenc}
  \RequirePackage{mathpazo}%
  \linespread{1.05}%
  \RequirePackage[scaled]{helvet}%
  \RequirePackage{courier} % tt
  \fi}%
{}
```

- Utopia
  Utopia is a transitional serif font designed by Robert Slimbach for Adobe in 1989. It became free software in 2006. It is paired here with Bera and Inconsolata, as is Garamond.

  Note that the utopia font clashes with the `amssymb` package.

  org-buffer-utopia() ≡

```
#+LaTeX_CLASS_OPTIONS: [utopia]
```

  option-utopia() ≡

```
\newboolean{utopia}
\DeclareOption{utopia}{\setboolean{utopia}{true}}
```

  load-utopia() ≡

```
\ifthenelse{\boolean{utopia}}
{%
  \ifpdf
   \RequirePackage[T1]{fontenc}
   \RequirePackage[adobe-utopia]{mathdesign}
   \RequirePackage[scaled]{berasans}
   \RequirePackage{inconsolata} % tt
  \fi}%
{}
```

- Charter
  Charter was designed to reproduce well on low-resolution 300 dpi printers. It is paired here with Helvetica and Courier, like Times, for which it is an alternative.

  These fonts conflict with the `amssymb` package.

  org-buffer-charter() ≡

```
#+LaTeX_CLASS_OPTIONS: [charter]
```

  option-charter() ≡

```
\newboolean{charter}
\DeclareOption{charter}{\setboolean{charter}{true}}
```

  Helvetica is set a bit smaller to better match the Charter font.

  load-charter() ≡

```
\ifthenelse{\boolean{charter}}
{%
  \ifpdf
   \RequirePackage[T1]{fontenc}
   \RequirePackage[bitstream-charter]{mathdesign}
   \RequirePackage[scaled=.90]{helvet}
   \RequirePackage{courier} % tt
  \fi}%
{}
```

1.5.4 Other packages

Packages not included in the Org-mode list of default packages are made available in `org-article.cls`. These include facilities to apply microtypographic adjustments to suitable fonts, set the line spacing of the document to double space, set lists more compactly than the standard LaTeX `article.cls`, and typeset source code listings using one of several color or black and white themes.

- Microtype
  The `microtype` package makes available the micro-typographic extensions of pdfTeX. Prominent among these are font expansion and character protrusion, which together result in fewer bad line breaks and a visually even right margin.

  You can probably read the `microtype` documentation, which runs to more than 200 pages, on your system by issuing the shell command:

  read-microtype() ≡

  ```
  texdoc microtype
  ```

  This package is not loaded by default. An option is provided to load it.

  org-buffer-microtype() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [microtype]
  ```

  option-microtype() ≡

  ```
  \newboolean{microtype}
  \DeclareOption{microtype}{\setboolean{microtype}{true}}
  ```

  load-microtype() ≡

```
  \ifthenelse{\boolean{microtype}}
{%
  \ifpdf
   \RequirePackage{microtype}
  \fi}%
{}
```

- Setspace
  The `setspace` package is used here for the sole purpose of creating double-spaced documents, such as manuscripts submitted to some publishing houses. If it is loaded, then the option `doublespace` will produce a double-spaced document.

  This package is not loaded by default. An option is provided to load it, and to set linespacing to doublespace.

  org-buffer-setspace() ≡

```
#+LaTeX_CLASS_OPTIONS: [setspace,doublespace]
```

  option-setspace() ≡

```
\newboolean{setspace}
\newboolean{doublespace}
\DeclareOption{setspace}{\setboolean{setspace}{true}}
\DeclareOption{doublespace}{\setboolean{doublespace}{true}}
```

  load-setspace() ≡

```
\ifthenelse{\boolean{setspace}}
{\RequirePackage{setspace}}
{}
```

  setspace-code() ≡

```
\ifthenelse{\boolean{setspace}}%
{\ifthenelse{\boolean{doublespace}}%
{\doublespacing}%
{\singlespacing}}%
{}%
```

- Paralist
  The `paralist` package was designed to meet the widespread request for

more tightly set lists in the standard LaTeX classes. If it is loaded, then the LaTeX environments `itemize`, `enumerate`, and `description` are over-ridden by their paralist counterparts.

You can probably read the `paralist` documentation on your system by issuing the shell command:

read-paralist() ≡

```
texdoc paralist
```

This package is not loaded by default. An option is provided to load it.

org-buffer-paralist() ≡

```
#+LaTeX_CLASS_OPTIONS: [paralist]
```

option-paralist() ≡

```
\newboolean{paralist}
\DeclareOption{paralist}{\setboolean{paralist}{true}}
```

load-paralist() ≡

```
% Set the standard LaTeX list environments to their compact
    counterparts
\ifthenelse{\boolean{paralist}}
  {%
    \RequirePackage{paralist}
    \let\itemize\compactitem%
    \let\description\compactdesc%
    \let\enumerate\compactenum%
  }
  {}
```

- Topcapt
  The `topcapt` package is needed when it is desired to set the caption of a table above the table. In this case, the `\caption{}` command must be moved above the `tabular` environment and the command changed to `\topcaption`. These will be adjustments made to the code produced by the Org-mode LaTeX exporter.

  This package is not loaded by default. An option is provided to load it.

  org-buffer-topcapt() ≡

```
#+LaTeX_CLASS_OPTIONS: [topcapt]
```

option-topcapt() ≡

```
\newboolean{topcapt}
\DeclareOption{topcapt}{\setboolean{topcapt}{true}}
```

load-topcapt() ≡

```
\ifthenelse{\boolean{topcapt}}
  {\RequirePackage{topcapt}}
  {}
```

- Listings
  The `listings` package is a source code printer for LaTeX. Except for the two options `draft` and `final`, which the `listings` package is configured to pick up itself from options passed to `\documentclass`, the other options were introduced to ease debugging or to trigger compatibility with earlier versions of the package. It seems unwise to use this mechanism to set options fo rthe `listings` package because there is no reason to assume that it will be stable. One solution would be to process options for this package using a `key = value` interface that sets the values of keys recognized by the package's `lstset` function. This is relatively difficult to do. An easier approach groups package options into themes, which can be selected with simple options, rather than `key = value` pairs. It is the approach adopted here.

  This package is not loaded by default. Options are provided to load it in its default state, set up for black and white reproduction, and set up for color reproduction.

  org-buffer-listings() ≡

  ```
  #+LaTeX_CLASS_OPTIONS: [listings, listings-bw, listings-
     color]
  ```

  This is where themes are defined for the listings package. The `listings-color` theme was lifted from a post to the Org-mode list by Eric Schulte. The `listings-sv` theme was posted to the list by Sebastian Vauban; it has been modified here to work with the `color` package, rather than the `xcolor` package used by Sebastian, and to allow breaking of long lines.

  option-listings() ≡

```
\newboolean{listings}
\newboolean{color}
\DeclareOption{listings}{\setboolean{listings}{true}}
\DeclareOption{listings-bw}{%
  \setboolean{listings}{true}%
  \AtBeginDocument{%
    \lstset{
      basicstyle=\ttfamily\footnotesize,%
      frame=lines,%
      breaklines=true,%
      showstringspaces=false}%
  }%
}
\DeclareOption{listings-color}{%
  \setboolean{listings}{true}%
  \setboolean{color}{true}%
  \AtBeginDocument{%
    \definecolor{keywords}{RGB}{255,0,90}%
    \definecolor{comments}{RGB}{60,179,113}%
    \definecolor{back}{RGB}{231,231,231}%
    \lstset{%
      keywordstyle=\color{keywords},%
      commentstyle=\color{comments},%
      backgroundcolor=\color{back},%
      basicstyle=\ttfamily\footnotesize,%
      showstringspaces=false,%
      frame=lines,%
      breaklines=true%
    }%
  }%
}
\DeclareOption{listings-sv}{%
  \setboolean{listings}{true}%
  \setboolean{color}{true}%
  \AtBeginDocument{%
    \definecolor{...@lstbackground}{RGB}{255,255,204} %
       light yellow
    \definecolor{...@lstkeyword}{RGB}{0,0,255} % blue
    \definecolor{...@lstidentifier}{RGB}{0,0,0} % black
    \definecolor{...@lstcomment}{RGB}{255,0,0} % red
    \definecolor{...@lststring}{RGB}{0,128,0} % dark green
    \lstset{%
      basicstyle=\ttfamily\scriptsize, % the font that is
          used for the code
      tabsize=4, % sets default tabsize to 4 spaces
```

```
      numbers=left, % where to put the line numbers
      numberstyle=\tiny, % line number font size
      stepnumber=0, % step between two line numbers
      breaklines=true, %!! do break long lines of code
      showtabs=false, % show tabs within strings adding
         particular underscores
      showspaces=false, % show spaces adding particular
         underscores
      showstringspaces=false, % underline spaces within
         strings
      keywordstyle=\color{...@lstkeyword},
      identifierstyle=\color{...@lstidentifier},
      stringstyle=\color{...@lststring},
      commentstyle=\color{...@lstcomment},
      backgroundcolor=\color{...@lstbackground}, % sets the
         background color
      captionpos=b, % sets the caption position to 'bottom'
      extendedchars=false %!?? workaround for when the
         listed file is in UTF-8
    }%
  }%
}
```

load-listings() ≡

```
\ifthenelse{\boolean{listings}}
  {\RequirePackage{listings}}
  {}
```

listings-code() ≡

```
\ifthenelse{\boolean{listings}}%
  {\lstdefinelanguage{org}
    {morecomment=[l]\#}%
  }
{}%
```