The **cd-cover** class[*]
Version v1.4 Beta

CVS Revision: 1.6

Christian Holm[†]

2002/01/28 14:30:44
CVS Date: 2002/01/28 14:30:44

Contents

---

[*] Previously a package

[†] E–mail: `cholm@nbi.dk`

1 Introduction

This is an β release of the cd-cover class. This was previously a package, but since a class is actually more appropiate for the commands and macros herein, this new version switches to a class implementation.

A number of improvements has been made in this release. The whole scheme of things has taken a drastic turn, providing many new advantages (see also section 5.2).

Bug reports, suggestions, praises, but not flames should be sent to `cholm@nbi.dk`. Please submit changes in form of patches to the most recent `cd-cover.dtx` file[1]. Patches are easily generated with

```
diff -c ⟨original-file⟩ ⟨new-file⟩ > ⟨patch-file⟩
```

on any reasonable system. This makes merging changes into the class much eaiser, and the changes will more like be included in the normal distribution.

2 Usage

In this release four environments are provided:

- bookletsheets is for typesetting booklet pages for normal CD covers (plastic cover).
- backssheet is for typesetting the back side for normal CD covers.
- backssheet* Special version of the above, that rotates the back title the other way. Useful for typesetting double jewelcase covers, and for some nationalities (e.g., German) that like the title to run opposite to that of English.
- sleevesheet is for typesetting paper sleeves for CDs, for example if you lost the plastic one, or don't like the space they take up, or whatever.
- singlesheet is for typeseting covers for plastic single CD covers.

2.1 Syntax

The syntax of these environments is:

```
\begin{bookletsheets}⟨your text⟩\end{bookletsheets}
\begin{backsheet}{⟨title⟩}⟨your text⟩\end{backsheet}
\begin{backsheet*}{⟨title⟩}⟨your text⟩\end{backsheet*}
```

---

[1] Avaliable from CTAN in `macros/latex/contrib/other/cd-cover`.

```
\begin{sleevesheet}⟨your text⟩\end{sleevesheet}
\begin{singlesheet}{⟨title⟩}{⟨slip text⟩}⟨your text⟩ \end{singlesheet}
```

Here `{⟨title⟩}` is the title on the back of the covers, `{⟨slip text⟩}` is the text that should go on the back slip of the single type cover.

In figure 1 to 4 there are some examples of these environments.

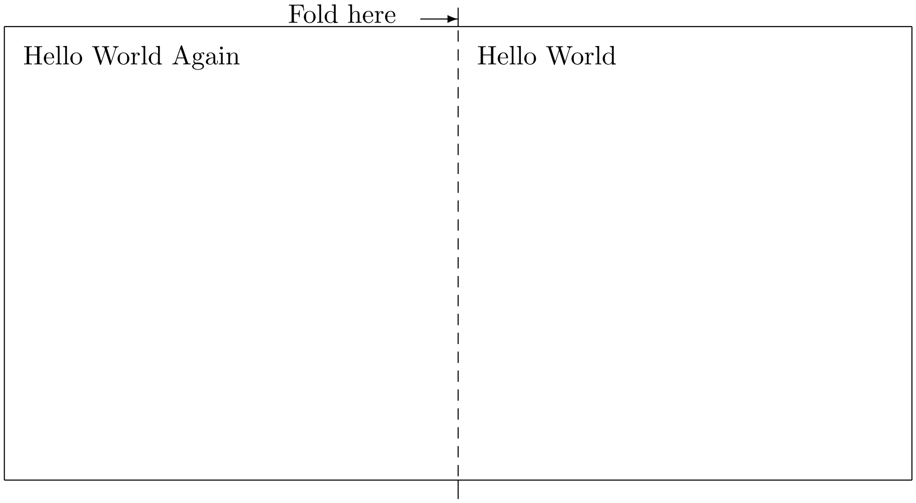

Figure 1: Example of bookletsheets output

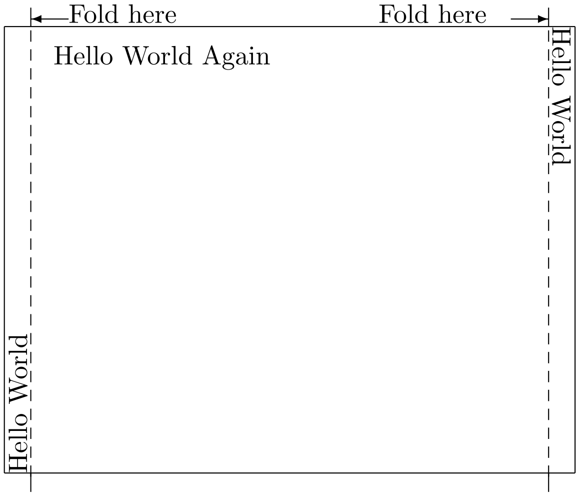

Figure 2: Example of backsheet output

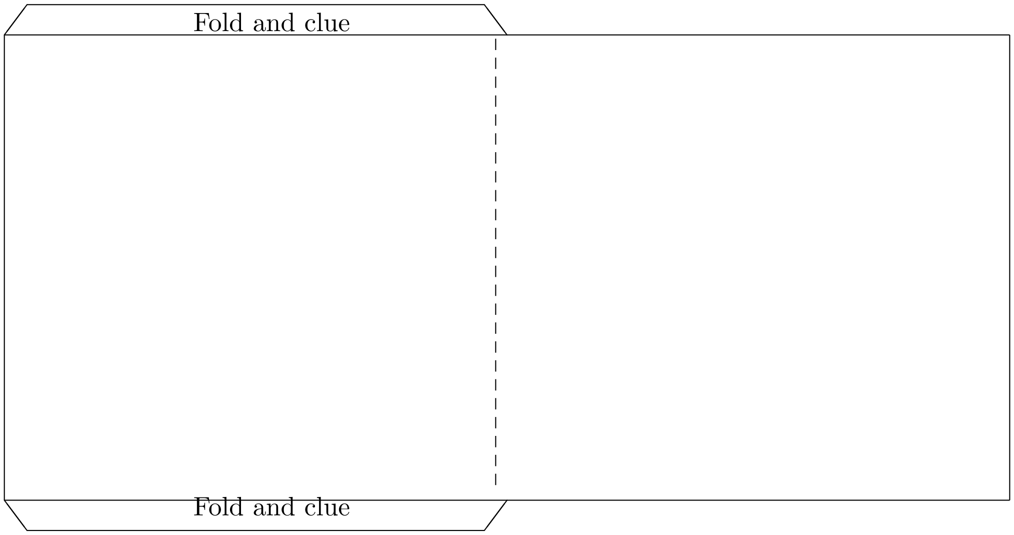

Figure 3: Example of sleevesheet output

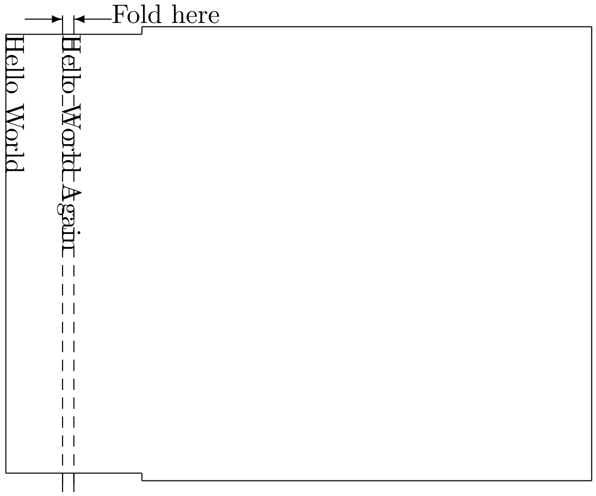

Figure 4: Example of singlesheet output

2.2 User setup

For each of the four environments, two user configurable lengths are defined. One is `\CD`⟨*type*⟩`TopMargin`, which determins the vertical space from the cover border to ⟨*your text*⟩. The other is `\CD`⟨*type*⟩`Margin` which determins the horizontal space between the border and
metayour text. The default values for these lengths are 5mm.

Here is the complete list of the lengths

`\CDbookletTopMargin`	`\CDbackTopMargin`
`\CDbookletMargin`	`\CDbackMargin`
`\CDsleeveTopMargin`	`\CDsingleTopMargin`
`\CDsleeveMargin`	`\CDsingleMargin`

For example, if you want to put a picture that fills out a complete page of the booklets, you can set `\CDbookletTopMargin` and `\CDbookletMargin` to `0pt`. For example

```
\documentclass{cd-cover}
\usepackage{epsfig}
\begin{document}
\parindent=0pt
\parskip=0pt
\CDbookletTopMargin=0pt
\CDbookletMargin=0pt
\begin{bookletsheets}
  \vspace{5mm}  % These are specific for the file rh61img.ps
  \hspace{5mm}
  \begin{center}
    {\LARGE Debian GNU/Linux 2.2r5 --- potato}
  \end{center}
  \epsfig{file=rh61img.ps,width=11.99cm}
  \end{bookletsheets}
\end{document}
```

Note that the graphics should not be 12cm exactly, but rahter slightly less, say 11.99cm like above. This is to make sure that it only takes up one page, so that LaTeX will not start a new page.

3 Credits

Many people have come up with suggestions, bugs, improvements to the first package version of `cd-cover`, and I'd like to thank them, especially for thier patience in me, for not getting this out sooner. So thanks to James A. Bednar, Steven Buehler, Patrik Carlsson, Jim Clark, Lionel Cons, Joerg Desch, Stefan A. Deutscher, Holger Dewes, Stephen Gildea, Anders Bruun Olsen, Rolf Niepraschk, Mark Peade, Dominik Roettsches, Carsten Schäfer, C. J. Walker, and Rainer Wiener. I may have forgotten some, for which I apologize.

And of course thanks to Donald E. Knuth for TeX, Leslie Lamport for LaTeX, Robin Fairbairns for CTAN and being very patient with me.

4 Copyright

5 Implementation

5.1 Initializing

Identification

First we must, as any good package or class should, identify ourselves, and tell what format we need. In this case it is ofcourse LaTeX 2ε.

```
\NeedsTeXFormat{LaTeX2e}
\def\fileversion{v1.4}
\def\filedate{2002/01/28}  % Format is YYYY/MM/DD
\ProvidesClass{cd-cover}[\filedate\space\fileversion
\space cd-cover class.]
```

Lengths or dimensions

The next thing is to setup some lengths (dimensions) we need for various purposes. There is two temporary lengths `\temp@length` and `\temp@@length`, for use in variuos calculations; along width margin lengths, two for each type of environment, one for the odd pages `\booklet@odd@margins`, `\back@odd@margins`, `\sleeve@odd@margins`, and `\single@odd@margins`, and ofcourse the corrosponding for the even pages `\booklet@even@margins`, `\back@even@margins`, `\sleeve@even@margins`, and `\single@even@margins`. These margin lengths is used so that we may do two–sided printing where the various covers align.

```
\newlength{\temp@length}
\newlength{\temp@@length}
\newlength{\booklet@even@margins}
\newlength{\booklet@odd@margins}
\newlength{\back@even@margins}
\newlength{\back@odd@margins}
\newlength{\sleeve@even@margins}
\newlength{\sleeve@odd@margins}
\newlength{\single@even@margins}
\newlength{\single@odd@margins}
```

Options

`\cd@cover@setup@margins`

This command sets up a number of dimensions, so that when doing two–side printing, the covers will be align on each side of the paper, and can neatly be cut out to provide a double–sided booklet, sleeve or cover. This is not implemeted fully yet. What is here is good for the booklet, but doesn't work for single other types of covers. It's mainly the horizontal alignment that's the problem.

```
\def\cd@cover@setup@margins{
  % This is some old stuff, that I'm not sure is needed, but I leave it
  % \global\paperheight=\temp@length
  % \global\paperwidth=\temp@@length
  \advance\temp@length-\booklet@height
  \divide\temp@length2
  \advance\temp@length-1in
  \advance\temp@length-\headheight
  \global\topmargin=\temp@length
  \advance\temp@@length-\booklet@width
  \divide\temp@@length2
  \advance\temp@@length-1in
  \oddsidemargin=\temp@@length}
\AtBeginDocument{\cd@cover@setup@margins}
```

We also provide a number of options, to specify the papersize. Notice, that since the article class will set the paper size, we use temporary local variables to hold the information, and then in `\cd@cover@setup@margins` we set the right dimensions.

```
\DeclareOption{a4paper}{
  \global\temp@length=210mm
  \global\temp@@length=297mm}
\DeclareOption{letterpaper}{
  \global\temp@length=8.5in
  \global\temp@@length=11in}
\DeclareOption{legalpapar}{
  \setlength\temp@length{8.5in}
  \setlength\temp@@length{14in}}
\DeclareOption{executivepaper}{
  \setlength\temp@length{7.25in}
  \setlength\temp@@length{10.5in}}
```

Notice, that some standard papersizes produces an error message, as they will be too small to hold the covers.

```
\DeclareOption{a5paper}{\cd@cover@paper@error{a5paper}}
\DeclareOption{b5paper}{\cd@cover@paper@error{b5paper}}
```

Also, all papersizes are already in ‘landscape’ version[2] so the landscape option is redundent with this class.

```
\DeclareOption{landscape}{
  \ClassWarning{cd-cover}{Option ‘landscape’ redundant to this %
    class.}}
```

Add an option to avoid foldlines

```
\newif\ifcd@cover@foldlines\cd@cover@foldlinestrue
\DeclareOption{nofoldlines}{\cd@cover@foldlinesfalse}
\DeclareOption{foldlines}{\cd@cover@foldlinestrue}
```

All other options are passed on to the standard article class, as expected. ‘letterpaper’ is then the default paper size, and we finally execute the options.

```
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
\ExecuteOptions{letterpaper}
\ProcessOptions\relax
```

`\cd@cover@paper@error` This is just an error message, so that the user doesn’t try to use a paper size that is too small to hold some of the covers.

```
\def\cd@cover@paper@error#1{
  \ClassError{cd-cover}{^^J%
    Paper format ‘#1’ is too small for the covers}{^^J%
    You have given the option ‘#1’ to this class.^^J%
    However, that paper size is too small to hold^^J%
    the covers.^^J%
    Please change the paper size option to another^^J%
    format that CAN hold the covers (e.g., ‘a4paper’,^^J%
    ‘letterpaper’, etc.)}}
```

---

[2] compared to the normal defintions in `article.cls`.

Class and package loading

First we load the standard LaTeX 2_ε class article with what ever options was passed from above. This provides the user with the standard envrionments and sectioning commands, and is therefore desired. The article class is chosen, since the sectioning commands of this class is appropiate for cd–covers. The package rotating is then loaded. This package facilitates rotation of text, using `\special` commands. This means, that this class may be sensible to the users PostScript driver. Also, as I myself suspected, and as others have confirmed, this package is flawed, meaning, that this class isn't as stable as it should be. However, the rotating package is the best avaliable for the job, and hence used here[3].

After loading the class, we set the paper height and width.

```
\LoadClass{article}
\paperheight=\temp@length
\paperwidth=\temp@@length
\RequirePackage{rotating}
```

Setting up fixed lengths

Next on the agenda is to define and setup some fixed lengths. These are the dimensions of the various cover styles. The really isn't much to say about these. I obtained these numbers by simple meassuring some of my own (pre–printed) covers, with, as you can see, a metric meter stick.

```
\newlength{\booklet@width}     \setlength{\booklet@width}{240mm}
\newlength{\booklet@height}    \setlength{\booklet@height}{120mm}
\newlength{\back@width}        \setlength{\back@width}{137mm}
\newlength{\back@height}       \setlength{\back@height}{118mm}
\newlength{\back@slip@width}   \setlength{\back@slip@width}{7mm}
\newlength{\sleeve@height}     \setlength{\sleeve@height}{128mm}
\newlength{\sleeve@width}      \setlength{\sleeve@width}{266mm}
\newlength{\single@height}     \setlength{\single@height}{120mm}
\newlength{\single@width}      \setlength{\single@width}{137mm}
\newlength{\single@slip@width} \setlength{\single@slip@width}{15mm}
\newlength{\single@back@width} \setlength{\single@back@width}{3mm}
```

Setting up user lengths

The user also has some control of the page dimensions of the various cover types. These are provided via the below lengths. What the do should be obvious from the names[4]. We also set the default values of these lengths here.

```
\newlength{\CDbookletTopMargin} \setlength{\CDbookletTopMargin}{5mm}
\newlength{\CDbookletMargin}     \setlength{\CDbookletMargin}{5mm}
```

---

[3] I'm sorry I don't have the time, or expertice, to write a new package, but that's life for you.

[4] Otherwise, refer to the documentation part of this document.

```
\newlength{\CDbackTopMargin}    \setlength{\CDbackTopMargin}{5mm}
\newlength{\CDbackMargin}       \setlength{\CDbackMargin}{5mm}
\newlength{\CDsleeveTopMargin}  \setlength{\CDsleeveTopMargin}{5mm}
\newlength{\CDsleeveMargin}     \setlength{\CDsleeveMargin}{5mm}
\newlength{\CDsingleTopMargin}  \setlength{\CDsingleTopMargin}{5mm}
\newlength{\CDsingleMargin}     \setlength{\CDsingleMargin}{5mm}
```

Finally we set the default page style to empty.

```
\pagestyle{empty}
```

5.2 The various cover types

The changing form one type of cover to the other is implemented using ‘page styles’. So each type of cover has it’s own page style. This has several advantages.

To those of you familiar with the first two (or was it three — I can’t remember) versions, this is where you really will note the difference. In version v1.0 from 1998/10/04, the interface was driven via commands. This limited a number of things, most noticably the text length. You had to manually break each page up, which ofcourse is very non–TeX–like. Secondly it limited the use of ‘floats’ such as tables and figures.

Well, in the *new* scheme, this is all changed. Since the pagestyle doesn’t effect the way pages are broken by TeX, one no longer has to break up the pages manually! You can put as much text as you like into the pages, LaTeX will just produce more pages as the text growes in length. Though page styles has *some* effect on the LaTeX 2ε output routine and hence on floats, it is very minimal, and floats can be use extensively under this scheme.

Another advantage of the new scheme, is the use of user *configurable dimensions*. If all the boundary dimensions are set to zero, you can effectually fill out the whole cover with text. Though this doesn’t seem usefull, consider the the case where you want to fill out your cover with some marvelous picture you have created in Gimp, PhotoShop, or just downloaded form the internet!

5.2.1 The booklet type

Let me just say a few words on page styles. Every page style defines the commands `\@oddhead`, `\@evenhead`, `\@oddfoot`, and `\@evenfoot` to be some command containg some text[5]. There *must* be some text present, or othetwise TeX will respond with a `Underfull \hbox... while output is active`, wich is quite enyoing. For this reason all the *border* macros contains a `\hfill` as the very last thing. Also, LaTeX assumes, that the four macros above are no wider then `\textwidth` and no taller then `\headheight` or `\footskip`, whatever appropiate.

For the page styles at hand, I decided to `\let` the above two `\@...head` commands

---

[5] In TeX terms this means *horizontal mode stuff.*

to the *border* commands below, and the two `\@...foot` to `\@empty`. Now these *border* commands are apparently *much* taller then `\headheight`, but if one takes a closer look, they actually aren't.

The first part of every one of them is one ore more `\kern`s. `\kern`s in TeX mind[6] doesn't take up space in anyway, so these commands are *invisble* in that sense. The next thing is a picture environment, and this is defined in LaTeX 2_ε so that it does't take up any space in the previous sense. Finishing of the macros is a `\hfill`, wich is there for reasons explained above. So the only command that is 'space visible' in the *border* commands, is the `\hfill`! Now if that isn't neat, I don't know what is. That said, I won't comment anymore on the page style definitons and move straight on to the rest of the macros and commands.

`\ps@bookletsheet`

```
\def\ps@bookletsheets{
  \let\@oddhead\booklet@border\let\@evenhead\@oddhead
  \let\@oddfoot\@empty\let\@evenfoot\@oddfoot}
```

`\init@booklet` We need to do some manipulation of the dimensions `\textheight`, `\textwidth`, `\headsep` and since the booklet pages are set in double column mode, also on `\columnsep` and `\@colht`.

```
\def\init@booklet{
```

What we do is to set the text height limit to the proper value for the booklet — as defined in `\booklet@height` — then correct for the space wanted above and below the text — as defined in `\CDbookletTopMargin`. This ofcourse only sets the *lower* margin, and has absolutly nothing to do with the upper margin. To do that we fix the value of `\headsep` to `\CDbookletTopMargin`.

```
  \temp@length=\booklet@height
  \advance\temp@length-2\CDbookletTopMargin
  \global\textheight=\temp@length
  \global\headsep=\CDbookletTopMargin
```

The next thing is to set the text width. This limit is again set to it's appropiate value for this type of cover — defined in `\booklet@width` — and then corrected for for the wanted margin width — taken from `\CDbookletMargin`.

```
  \temp@length=\booklet@width
  \advance\temp@length-2\CDbookletMargin
  \global\textwidth=\temp@length
```

A little extra care is needed since we type set in two column mode. We In the case, that the first page contains this sort of cover, we *have to* set `\@colht` equal to `\textheight` manually.

---

[6] Or mouth or stomach, I don't remember wich one it is, as Donald E. Knuth calls it in the TeXbook.

The reason is a bit complicated, and I will not go into much detail about it. The point is, that LaTeX uses `\@colht` as the limit for the height of the columns. Now, when constructing the first column, `\@colht` still has the old value of `\textheight`, as set in the begining of the document, and this is properly not what we want. *After* making the first column, it sets `\@colht` to the present value of `\textheight`, which is what we want, but then it is too late. The *only* way to get around this problem[7], is to manually set `\@colht`. One should not be conserned on that this dimension is hand–set on subsequent pages.

The second thing to do, is to set `\columnsep` to whatever value the margins should have — as defined in `\CDbookletMargin`.

```
  \global\@colht=\textheight
  \global\columnsep=2\CDbookletMargin}
```

`bookletsheets` Now for the environment it self. We start off with initializing the dimensions as outlined above, and then we start the two column mode and set the page style to bookletsheets. To finish of the the environment, we make sure that the last page has a border by calling `\clearpage`, and revert to the default page style.

```
\newenvironment{bookletsheets}{
  \init@booklet\twocolumn\pagestyle{bookletsheets}}{
  \clearpage\pagestyle{empty}}
```

`\booklet@border` First we want to back up the margin width (`\CDbookletMargin`), so that the border will be properly spaced from the text. Secondly we want to draw the border.

This border isn't very complex. It's just a retangular with to small lines indicating a folding line. For the users convinience, we put an arrow an some text next to the folding line.

As explained above this command is 'space invisible', execpt for the final `\hfill`, which is exactly what we want.

```
\def\booklet@border{%
  \kern-\CDbookletMargin%
  \setlength{\unitlength}{1mm}%
  \begin{picture}(100,1)%
    \put(0,0){\line(1,0){240}}%     Top most horizontal line
    \put(0,0){\line(0,-1){120}}%    Left most vertical line
    \put(0,-120){\line(1,0){240}}%  Bottom most horizontal line
    \put(240,0){\line(0,-1){120}}%  Right most vertical line
    \put(120,0){\line(0,1){5}}%     Top folding mark
    \put(120,-120){\line(0,-1){5}}% Bottom folding mark
    \ifcd@cover@foldlines
      \multiput(120,-1)(0,-5){24}{\line(0, -1){3}} % Fold line
    \fi
    \put(110,2){\vector(1,0){10}}%  Arrow
```

---

[7] Or is ts a bug in LaTeX 2ε.

```
  \put(95,1){Fold here}%          Helping text
\end{picture}\hfill}
```

5.2.2 The back type

One could argue, that the page style scheme outlined on page 10 really isn't that advantages in this case, and that a minipage–like solution would be better, since the text in this environment shouldn't be more then *one* page, since there is only room for one back sheet on a normal cd-cover. While this is true, I decided, that it should be up to the user to restain him/her–self from putting tom much text into these kinds of pages.

`\ps@backsheet` Page style for the backpage.

```
\def\ps@backsheet{
  \let\@oddhead\back@border\let\@evenhead\@oddhead
  \let\@oddfoot\@empty\let\@evenfoot\@empty}
```

`\init@back` This macro is similar to `\init@booklet` comment above, so I will suffice to say the instead of `\CDbooklet...` above read `\CDback...`, and the two–column notes are irrelevant.

```
\def\init@back{
  \temp@length=\back@height
  \advance\temp@length-2\CDbackTopMargin
  \global\textheight=\temp@length
  \temp@length=\back@width
  \advance\temp@length-2\CDbackMargin
  \global\textwidth=\temp@length
  \global\headsep=\CDbackTopMargin}
```

`\back@title` This is a save box for the title on the back of the cover. See more below.

```
\newsavebox{\back@title}
```

`backsheet`
`backsheet*` This environment should take one argument, the title on the back of the cover. This will be type set two times; one on each end of the cover, rotated as to give the proper reading when your CD is put on the shelf. The argument is put into a `\parbox`, so that we may control the size of the text, and then put into a save box for later use (in `\back@border`).

The star'ed version backsheet* turns the back title opposite to the non–star'ed version. This is useful if one want's to typeset covers for double jewel cases. Also, in some countries, like Germany, they like having the text run the opposite direction, so for those users, that environment may be more appropiate.

`\ifback@anglereverse` To help distingish between the two ways of rotation the back title, we have the macro `\ifback@anglereverse`, which is set to `\relax` if we're using the regular non–star'ed version.

```
\newif\ifback@anglereverse\back@anglereversefalse
```

As in bookletsheets environment, we call the initializing macro, make sure we are in one–cloumn mode, and set the page style. When finishing of the environment, we clear the page, to insure the last page style, and revert to the default page style.

```
\newenvironment{backsheet}[1]{
  \back@anglereversefalse%
  \init@back%
  \savebox{\back@title}[\textheight]{%
    \parbox[t][6mm]{\textheight}{#1}}
  \onecolumn\pagestyle{backsheet}
  \clearpage}{\clearpage\pagestyle{empty}}
\newenvironment{backsheet*}[1]{
  \back@anglereversetrue%
  \init@back%
  \savebox{\back@title}[\textheight]{%
    \parbox[t][6mm]{\textheight}{#1}}
  \onecolumn\pagestyle{backsheet}
  \clearpage}{\clearpage\pagestyle{empty}}
```

`\back@border` First we back up, just as in `\booklet@border`, but we also have to back up the length (height) of the slips, unlike above. Then the border is drawn with folding marks, and some help text and arrows.

```
\def\back@border{%
  \kern-\CDbackMargin%
  \kern-\back@slip@width%
  \setlength{\unitlength}{1mm}%
  \begin{picture}(0,0)%
    \put(0,0){\line(1,0){151}}%     Top most horizontal line
    \put(0,-118){\line(1,0){151}}%  Bottom most horizontal line
    \put(0,0){\line(0,-1){118}}%    Left most vertical line
    \put(151,0){\line(0,-1){118}}%  Right most vertical line
    \put(7,0){\line(0,1){5}}%       Left top folding mark
    \put(7,-118){\line(0,-1){5}}%   Left bottom folding mark
    \ifcd@cover@foldlines
      \multiput(7,-1)(0,-5){24}{\line(0, -1){3}} % Fold line
    \fi
    \put(144,0){\line(0,1){5}}%     Right top folding mark
    \put(144,-118){\line(0,-1){5}}% Right bottom folding mark
    \ifcd@cover@foldlines
      \multiput(144,-1)(0,-5){24}{\line(0, -1){3}} % Fold line
    \fi
    \put(17,2){\vector(-1,0){10}}%  Left help arrow
    \put(17,1){Fold here}%          Left help text
    \put(134,2){\vector(1,0){10}}%  Right help arrow
    \put(119,1){Fold here}%         Right help text
```

We finshes off with drawing the rotated titles. Please notice, that the left one, is the one closest to the opening of the CD cover. This should explain the somewhat strange rotating values.

Also, notice that we can rotate one or the other way, as explained above, conditionally on `\ifback@anglereverse{1}`.

```
    \ifback@anglereverse%
      \put(1,0){\begin{rotate}{-90}\usebox{\back@title}\end{rotate}}%
      \put(150,-118){\begin{rotate}{90}\usebox{\back@title}\end{rotate}}%
    \else%
      \put(6,-118){\begin{rotate}{90}\usebox{\back@title}\end{rotate}}%
      \put(145,0){\begin{rotate}{-90}\usebox{\back@title}\end{rotate}}%
    \fi%
    %\put(3,-118){\begin{rotate}{90}\usebox{\back@title}\end{rotate}}%
    %\put(148,0){\begin{rotate}{-90}\usebox{\back@title}\end{rotate}}%
  \end{picture}\hfill}
```

5.2.3 The sleeve type

This type showes how you can juggle around with LaTeX and get it to do almost everything you want[8]. By altering a bit in the way `\LaTeXe` output two–column pages, I could get it to swap the columns, so that the last entered would come out first, and vice versa. This is exactly what we want for this page style, which is obvious when you see the finished output.

`\ps@sleevesheet`

```
\def\ps@sleevesheet{%
  \let\@oddhead\sleeve@border\let\@evenhead\@oddhead
  \let\@oddfoot\@empty\let\@evenfoot\@oddfoot}
```

`\init@sleeve` What I said about `\init@booklet` goes here with the exchange of `\CDsleeve...` for `\CDbooklet....`

```
\def\init@sleeve{
  \temp@length=\sleeve@height
  \advance\temp@length-2\CDsleeveTopMargin
  \global\textheight=\temp@length
  \global\@colht=\textheight
  \temp@length=\sleeve@width
  \advance\temp@length-2\CDsleeveMargin
  \global\textwidth=\temp@length
  \global\headsep=\CDsleeveTopMargin
  \global\columnsep=2\CDsleeveMargin}
```

---

[8] I don't want to sound pretentious, but I was surprised how easily it all turned out to be.

`sleevesheet` Again we initialize, make sure we are in two column mode, and set the page style. The only thing to notice here, is we `\let \@outputdblcol` to `\sleeve@outputdblcol`, and when the environment ends, change back to the usual definition saved in `\ltx@outputdblcol`. It is important that the last change comes *after* the `\clearpage`, since otherwise, the last page will be output with the wrong output routine. Notice, that it doesn't make sense to make two–side prinint for this kind of cover, so the scheme works on *any* page. Ofcourse, you should make the following page blank, by putting `\vfill\clearpage` after the environment.

```
\newenvironment{sleevesheet}{%
  \init@sleeve
  \twocolumn\pagestyle{sleevesheet}
  \global\let\@outputdblcol\sleeve@outputdblcol}{
  \clearpage\pagestyle{empty}
  \global\let\@outputdblcol\ltx@outputdblcol}
```

`\sleeve@border` Two flaps is outlined for clueing.

```
\def\sleeve@border{%
  \kern-\CDsleeveMargin%
  \setlength{\unitlength}{1mm}%
  \begin{picture}(100,1)%
    \put(0,0){\line(3,4){6}}%        Top flap
    \put(6,8){\line(1,0){121}}%
    \put(133,0){\line(-3,4){6}}%
    \ifcd@cover@foldlines
      \multiput(133,-1)(0,-5){25}{\line(0, -1){3}} % Fold line
    \fi
    \put(50,1){Fold and clue}%       Help text in top flap
    \put(0,-123){\line(3,-4){6}}%    Bottom flap
    \put(6,-131){\line(1,0){121}}%
    \put(133,-123){\line(-3,-4){6}}%
    \put(50,-131){Fold and clue}%    Help text in bottom flap
    \put(0,0){\line(1,0){266}}%      Top most horizontal line
    \put(0,0){\line(0,-1){123}}%     Left most vertical line
    \put(0,-123){\line(1,0){266}}%   Bottom most horizontal line
    \put(266,0){\line(0,-1){123}}%   Right most vertical line
  \end{picture}\hfill}
```

5.2.4 The single type

`\ps@singlesheet` To facilitate two side printing, we have to have two kinds of borders, since the border outline is asymmetric along a vertical line, hence `\@oddhead` and `\@evenhead` has different values.

```
\def\ps@singlesheet{
  \let\@oddhead\odd@single@border
  \let\@evenhead\even@single@border
  \let\@oddfoot\@empty\let\@evenfoot\@oddfoot}
```

`\single@title`
`\single@slip`

These two save boxes are used below.

```
\newsavebox{\single@title}
\newsavebox{\single@slip}
```

`\init@single`

Again exchnage `\CDsingle`... with `\CDback`... in the description of `\init@back`, and there you have it.

```
\def\init@single{
  \temp@length=\single@height
  \advance\temp@length-2\CDsingleTopMargin
  \advance\temp@length-4mm
  \global\textheight=\temp@length
  \temp@length=\single@width
  \advance\temp@length-2\CDsingleMargin
  \global\textwidth=\temp@length
  \global\headsep=\CDsingleTopMargin}
```

`singlesheet`

As in the backsheet environment, we save the arguments in `\parbox`'es to control the size, and put all that into save boxes, wich are used in `\odd@single@border`. Everything else is as above.

```
\newenvironment{singlesheet}[2]{
  \init@single
  \savebox{\single@title}[\textwidth]{%
    \parbox[t][3mm]{\textwidth}{#1}}
  \savebox{\single@slip}[\textwidth]{%
    \parbox[t][\single@slip@width]{\textwidth}{#2}}
  \onecolumn\pagestyle{singlesheet}
  \clearpage}{\clearpage\pagestyle{empty}}
```

`\odd@single@border`

We define two border for the reasons stated above. The differ in that they are mirror reflections of each other, and the back title and slip text isn't put on even sides.

```
\def\odd@single@border{
  \kern-\CDsingleMargin%
  \kern-\single@slip@width%
  \kern-\single@back@width%
  \setlength{\unitlength}{1mm}%
  \begin{picture}(0,0)%
    \put(0,0){\line(0,-1){116}}%                  Left most vertical line
    \put(0,0){\line(1,0){36}}%                    First top horizontal line
    \put(36,0){\line(0,1){2}}%                    Jump
    \put(15,0){\line(0,1){5}}%                    Fold mark
    \ifcd@cover@foldlines
      \multiput(15,-1)(0,-5){24}{\line(0, -1){3}} % Fold line
    \fi
    \put(18,0){\line(0,1){5}}%                    Fold mark
    \ifcd@cover@foldlines
      \multiput(18,-1)(0,-5){24}{\line(0, -1){3}} % Fold line
    \fi
    \put(36,2){\line(1,0){119}}%                  Second top line
    \put(155,2){\line(0,-1){120}}%                Right most vertical line
    \put(0,-116){\line(1,0){36}}%                 First bottom line
    \put(36,-116){\line(0,-1){2}}%                Jump
    \put(15,-116){\line(0,-1){5}}%                Fold Mark
    \put(18,-116){\line(0,-1){5}}%                Fold Mark
    \put(36,-118){\line(1,0){119}}%               Second bottom line
    \put(5,4){\vector(1,0){10}}%                  Arrows
    \put(28,4){\vector(-1,0){10}}%                Arrows
    \put(28,3){Fold here}%                        Help text
    \put(0,0){\begin{rotate}{-90}\usebox{\single@title}\end{rotate}}%
    \put(15,0){\begin{rotate}{-90}\usebox{\single@slip}\end{rotate}}%
  \end{picture}\hfill}
```

`\even@single@border`

```
\def\even@single@border{
  \kern-\CDsingleMargin%
  \setlength{\unitlength}{1mm}%
  \begin{picture}(0,0)%
    \put(0,2){\line(0,-1){120}}%    1st left vert line
    \put(119,0){\line(1,0){36}}%    2nd top horiz line
    \put(119,0){\line(0,1){2}}%     2nd top right vert line
    \put(137,0){\line(0,1){5}}%     1st top fold mark
    \put(140,0){\line(0,1){5}}%     2nd top fold mark
    \put(0,2){\line(1,0){119}}%     1st top horiz line
    \put(155,0){\line(0,-1){116}}%  1st right vert line
    \put(119,-116){\line(1,0){36}}% 2nd bototm horiz line
    \put(119,-116){\line(0,-1){2}}% 2nd bottom right vert line
    \put(137,-116){\line(0,-1){5}}% 1st bottom fold mark
    \put(140,-116){\line(0,-1){5}}% 2nd bottom fold mark
    \put(0,-118){\line(1,0){119}}%  1st bottom horiz line
    \put(127,4){\vector(1,0){10}}%  1st help arrow
    \put(150,4){\vector(-1,0){10}}% 2nd help arrow
    \put(112,3){Fold here}%         Help text
  \end{picture}\hfill}
```

5.3 Modified output routines

Here are some modifications to the LaTeX 2$_{\varepsilon}$ output routine to *invert* the page orders in some of the environments, such as sleeve. First we save the old definition in `\ltx@outputdblcol`.

```
\let\ltx@outputdblcol\@outputdblcol
```

`\sleeve@outputdblcol`

Then we go on to define `\sleeve@outputdblcol`, changing the order of column output.

The first part os the same as in `ltoutput.dtx`, so no need to comment on that here.

```
\def\sleeve@outputdblcol{%
  \if@firstcolumn
    \global\@firstcolumnfalse
    \global\setbox\@leftcolumn\box\@outputbox
  \else
    \global\@firstcolumntrue
```

Now for the building of the actual output box, we *interchange* the order of `\@leftcolumn` and `\@outputbox` in the body definition of `\@outputbox`, tricky isn't it!

```
    \setbox\@outputbox\vbox{%
      \hb@xt@\textwidth{%
        \hb@xt@\columnwidth{%
          \box\@outputbox\hss}%  \@leftcolumn changed to \@outputbox
        \hfil\vrule\@width\columnseprule\hfil
        \hb@xt@\columnwidth{%
          \box\@leftcolumn\hss}}}% \@outputbox changed to \@leftcolumn
    \@combinedblfloats
    \@outputpage
    \begingroup
      \@dblfloatplacement
      \@startdblcolumn
      \@whilesw\if@fcolmade\fi
        {\@outputpage
          \@startdblcolumn}%
    \endgroup
  \fi
}
```

5.4 The final stuff

If we can figure out the graphics driver automatically, then we emit a `\special` or equivilant command to the output file. If we can not find the driver, we warn the user that the printing should be done in landscape mode.

```
\def\cdcover@endmessage{cd-cover class: Remember that this document is
    type set in landscape^^J mode, and therefore dvips should be
    passed the `-t landscape' option.}
\@ifundefined{Gin@driver}{%
  \AtEndDocument{\typeout{\cdcover@endmessage}}}{%
  \filename@parse{\Gin@driver}%
  \def\reserved@a{dvips}%
  \ifx\filename@base\reserved@a\relax%
  \message{DVIPS driver found}
  \AtBeginDvi{\special{papersize=\the\paperwidth,\the\paperheight}}%
  \AtBeginDvi{\special{papersize=\the\paperheight,\the\paperwidth}}%
\else
  \def\reserved@a{pdftex}\relax%
  \ifx\filename@base\reserved@a
    \message{PDF driver found}
    \pdfpagewidth=\the\paperwidth\pdfpageheight=\the\paperheight%
  \else
    \def\reserved@a{vtex}\relax%
    \ifx\filename@base\reserved@a
      \message{VTeX driver found}
      \mediawidth=\the\paperwidth\mediaheight=\the\paperheight%
    \else
      \AtEndDocument{\typeout{\cdcover@endmessage}}
    \fi
  \fi
\fi}
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.