

The atbegshi package

Heiko Oberdiek
<heiko.oberdiek at gmail.com>

2010/03/01 v1.11

Abstract

This package is a modern reimplementaion of package `everyshi` without the burden of compatibility. It makes use of ε -TeX's if available. Both L^AT_EX and plain-T_EX are supported.

Contents

1	Documentation	2
1.1	Examples	3
1.1.1	Example: circle in background	3
1.1.2	Example: adding TrimBox for dvipdfmx	4
2	Method of <code>\shipout</code> overloading	5
2.1	<code>\shipout</code>	5
2.2	<code>\afterassignment</code>	5
2.3	Test for direct or indirect boxes	6
2.3.1	With ε -TeX	6
2.3.2	Without ε -TeX	6
2.3.3	<code>\lastkern</code> method	7
2.4	Output	8
2.5	Separate box register	8
2.6	Summary	8
2.6.1	With ε -TeX	8
2.6.2	Without ε -TeX, traditional way	9
2.6.3	<code>\lastkern</code> method	9
3	Implementation	10
3.1	Reload check and package identification	10
3.2	Catcodes	11
3.3	Preparations	12
3.4	Positioning	15
3.5	Patches	17
3.5.1	Package <code>crop</code>	17
3.5.2	Package <code>everyshi</code>	19
3.5.3	Class <code>memoir</code>	20
4	Test	21
4.1	Catcode checks for loading	21
5	Installation	25
5.1	Download	25
5.2	Bundle installation	25
5.3	Package installation	26
5.4	Refresh file name databases	26
5.5	Some details for the interested	26

6 History	27
[2007/04/17 v1.0]	27
[2007/04/18 v1.1]	27
[2007/04/19 v1.2]	27
[2007/04/26 v1.3]	27
[2007/04/27 v1.4]	27
[2007/06/06 v1.5]	27
[2007/09/09 v1.6]	27
[2008/07/18 v1.7]	27
[2008/07/19 v1.8]	28
[2008/07/31 v1.9]	28
[2009/12/02 v1.10]	28
[2010/03/01 v1.11]	28
7 Index	28

1 Documentation

Package `atbegshi` redefines `\shipout` to insert hooks for user code that is executed before the page is shipped out. The code may modify or even discard the output page. Three hooks are implemented:

1. A hook that is executed for every page, see `\AtBeginShipout`
2. A hook that is executed for the next page only, see `\AtBeginShipoutNext`
3. A hook that is only executed for the first page, see `\AtBeginShipoutFirst`

The hooks are executed in this order. The following three macros provide the user interface for adding code to these hooks:

`\AtBeginShipout {<code>}`

Execute the `<code>` for every page. The page contents is held in box register `\AtBeginShipoutBox` and may be modified. Use `\AtBeginShipoutDiscard` if you want to discard the page.

Note: Package `everyshi` uses box register 255. With package `atbegshi` you must use `\AtBeginShipoutBox` instead.

If `LATEX` calls `\shipout` in `\@outputpage` (part of its output routine), the meaning of `\protect` is `\noexpand`. `LATEX` sets `\protect` to the appropriate `\@typeset@protect` in the box that is shipped out. This is too late for the hooks, they are called earlier in the redefined `\shipout`. Therefore package `atbegshi` sets `\protect` to `\@typeset@protect` before it calls the hooks. (In `\EveryShipout` of package `everyshi` the user is responsible for the correct setting of `\protect`.)

`\AtBeginShipoutNext {<code>}`

This reimplements package `everyshi`'s `\AtNextShipout`. The `<code>` is executed at shipout time of the next page only. It is just a convenience macro, it can be easily replaced by something like:

```
\newcommand{\MyShipoutHook}{}%
\AtBeginShipout{\MyShipoutHook}
\gdef\MyShipoutHook{%
... do something with next page ...}
```

```

\gdef\MyShipoutHook{}%
}

```

(This can be necessary, if hook order does matter).

`\AtBeginShipoutFirst {<code>}`

This reimplements L^AT_EX's `\AtBeginDvi`. This hook is usually used for `\special` commands that include PostScript header files. The `\code` is directly executed in a `\vbox` that is put at the beginning of the output page. Dealing with the output box `\AtBeginShipoutBox` is not necessary and not permitted here.

`\AtBeginShipoutDiscard`

This macro notifies package `atbegshi` that the output page is discarded. The remaining hook code and the remaining hooks are not executed and the page is thrown away. Also `\deadcycles` is cleared to zero like an ordinary `\shipout` would do.

`\AtBeginShipoutInit`

Usually the redefinition of `\shipout` is delayed by `\AtBeginDocument` (if this macro exists). This can be too late, if other packages also redefines `\shipout` and the order does matter. `\AtBeginShipoutInit` forces the immediate redefinition of `\shipout`.

`\AtBeginShipoutUpperLeft {<background material>}`

This is a macro that puts material in the background of box `\AtBeginShipoutBox`. The *<background material>* is set in an `\hbox`, the reference point is the upper left corner of the output page. In case of pdfL^AT_EX in PDF mode, the settings of `\pdfhorigin` and `\pdfvorigin` are respected.

The macro `\AtBeginShipoutUpperLeft` is intended to be used in one of the hook setting macros, such as `\AtBeginShipout`, `\AtBeginShipoutFirst`, or `\AtBeginShipoutNext`.

For L^AT_EX users the *<background material>* is set inside a `picture` environment:

```

\begin{picture}(0,0)
  \setlength{\unitlength}{1pt}%
  <background material>
\end{picture}

```

`\AtBeginShipoutUpperLeftForeground {<foreground material>}`

See `\AtBeginShipoutUpperLeft`. The difference is that the material is put in the foreground.

`\AtBeginShipoutOriginalShipout {<box>}`

It stores the meaning of `\shipout` at the time this package is loaded.

1.1 Examples

1.1.1 Example: circle in background

In this example we put a circle in the background in the middle of the paper.

```

1 (*example1)
2 \documentclass[a4paper]{article}
3 \usepackage{color}
4 \usepackage{atbegshi}

Package picture makes life a little easier, because we can now also use length
specifications in picture's commands.

5 \usepackage{picture}

Now we draw the circle in the middle of the paper. \put moves downwards,
because the origin is at the top of the page, not at its bottom.

6 \AtBeginShipout{%
7   \AtBeginShipoutUpperLeft{%
8     \put(0.5\paperwidth,-0.5\paperheight){\circle{10}}%
9   }%
10 }
11 \begin{document}
12 \section{Hello World}
13 \newpage
14 \AtBeginShipoutNext{%
15   \AtBeginShipoutUpperLeft{%
16     \color{red}%
17     \put(0,-0.5\paperheight){\line(1,0){\paperwidth}}%
18     \put(0.5\paperwidth, 0){\line(0,-1){\paperheight}}%
19   }%
20 }
21 Only on this page we add a red cross.
22 \newpage
23 This page has the circle only.
24 \par
25 \vspace{\fill}
26 The next page will be discarded.
27 \newpage
28 \AtBeginShipoutNext{%
29   \AtBeginShipoutDiscard
30 }
31 This page is discarded.
32 \newpage
33 The last page.
34 \end{document}
35 </example1>

```

1.1.2 Example: adding TrimBox for dvipdfmx

Now an example from “real life” follows. Someone from the mailing list for dvipdfmx wants to put a TrimBox on every page. If we use \AtBeginShipout, we have to put the \special inside the box \AtBeginShipoutBox that gets shipped out.

```

36 (*example2)
37 \documentclass{minimal}
38 \usepackage{atbegshi}
39 \usepackage[
40   dvipdfm,
41   paperwidth=630bp,
42   paperheight=810bp
43 ]{geometry}
44 \AtBeginShipout{%
45   \setbox\AtBeginShipoutBox=\hbox{%
46     \special{pdf: put @thispage <</TrimBox[9 9 621 801]>>}}%
47   \box\AtBeginShipoutBox
48 }%
49 }

```

```

50 \begin{document}
51   First page
52   \newpage
53   Second page
54 \end{document}
55 \end{example2}

```

Remember, in `\AtBeginShipoutBoxFirst` the `\setbox` wrapper code is implicitly given and the `\special` is used directly.

2 Method of `\shipout` overloading

2.1 `\shipout`

The TeX primitive command `\shipout` takes a box specification and puts the box as a new page in the output file. There are two kinds of box specifications:

Direct boxes: They are given by `\hbox`, `\vbox`, or `\vtop`,
e.g. `\shipout\hbox{Hello World}`.

Indirect boxes: `\box` or `\copy` references a box register by number. The box register contains the contents of the box.

Note: `\box` also clears the box register globally.

Then we have to differentiate between void and empty boxes:

Void: Initially or after `\box` there is no box in the box register. In this cases the box register is not empty, but *void*.

Empty: A box with empty contents, such as `\hbox{}` (`= \null`) or `\vbox{}` is an *empty hbox* or *empty vbox*. If a box register holds such a box, the box still exists, therefore the box register is *not void*.

2.2 `\afterassignment`

We want to overload `\shipout` to do something with the box. It is quite impossible to do this reliable by catching the box using macro arguments. The variety of box specifications is too large, Examples:

```

\shipout\null
\shipout\vbox{...}
\shipout\vtop\bgroup ... \egroup
\shipout\box255

```

Even worse, the braces don't need to be balanced:

```

\shipout\hbox\bgroup}
\shipout\vbox{\egroup

```

Happily TeX provides a reliable way via `\afterassignment`. It takes a macro name and executes it just after the assignment.

Now we can redefine `\shipout`. The box specification that follows `\shipout` is caught by `\setbox`. This is an assignment to a box register. `\afterassignment` notifies TeX, that we want to call `\@test` right after the assignment:

```

\shipout :=
  \afterassignment\@test
  \setbox\mybox=

```

We have seen different box specifications. Indirect boxes are easy to understand:

```

\shipout\box0 ⇒ \setbox\mybox=\box0 \boxed{\@test}

```

However direct boxes can have arbitrary contents with lots of other assignments. It would be quite unpredictable if `\TeX` would put `\@test` after the first of such an assignment or after the box specification if the box lacks of assignments. Therefore `\TeX` puts `\@test` right at the beginning of the box specification, e.g:

```
\shipout\hbox{Hello World}
⇒ \setbox\mybox=\hbox{\@test Hello World}
```

2.3 Test for direct or indirect boxes

Now we want to execute `\@test`, but where are we? We can be after the completed box assignment, if `\shipout` was called with an indirect box. Or we are right at the beginning of a direct box.

2.3.1 With ε -`\TeX`

With the ε -`\TeX`'s extensions the answer is very easy: Being inside the direct box means that we are inside a new group. The new primitive command `\currentgrouplevel` tells how deeply the groups are currently nested. Macro `\@test` just compares the previously stored group level with the current one:

```
\shipout :=
  \edef\saved@grouplevel{\number\currentgrouplevel}
  \afterassignment\@test
  \setbox\mybox=

\@test :=
  \ifnum\saved@grouplevel=\currentgrouplevel
    % case: indirect box, the assignment is completed
    \@output
  \else
    % case: direct box, we are inside the box
    \aftergroup\@outbox
  \fi
```

2.3.2 Without ε -`\TeX`

Life becomes complicate without ε -`\TeX`. We cannot ask the group level. However, if we are inside a direct box, the box register `\mybox` is not yet changed by `\setbox`. Thus we need a special initial value and compare it in `\@test` with the current value of the box.

What can be used as initial value? Arbitrary box contents cannot be compared. `\TeX` only tells us a few properties:

- Box type: `\ifhbox`, `\ifvbox`
- Dimensions: `\wd`, `\ht`, `\dp`
- Voidness: `\ifvoid`

Unhappily all these qualities even combined are not sufficient for constructing an initial box value, because `\shipout` can be called with a box that is accidentally just the same as the choosen initial value.

Nevertheless we have two alternatives for an initial value:

- A box of some type with some funny settings that are unlikely to occur in real life, e.g a height of `4911sp-\maxdimen`.
- A void box.

A collision between this initial value and an indirect `\shipout` box with just the same value is possible. Then `\@test` will make a wrong decision that it is executed inside a direct box and delays `\@output` by `\aftergroup`. Thus `\@output` is not called at the place we want. In contrary, the result is an uncertainty about the place:

- `\shipout` is used in a group that perhaps closes some pages later. A bad place for `\@output`.
- Without a surrounding group `\aftergroup` effectively kills its argument.

In the first case of a box with special dimensions we can even loose the page. However in the case of the void box, this effect is even desired, because the original `\shipout` does not output void boxes. All we have to do is to ensure that our box `\mybox` is always void except for the phase when the overloaded `\shipout` is executed. And secondly we must keep this semantics of `\shipout` for the void case in our macros, namely `\@output`.

```
\shipout :=
% trick to get a void box \mybox
\begingroup
\setbox\mybox=\box\mybox
\endgroup
\afterassignment\@test
\setbox\mybox=

\@test :=
\ifvoid\mybox
\aftergroup\@output
\else
\@output
\fi
```

The nasty case is `\shipout\box\voidb@x` where the indirect box is void and that must not generate an output page. If a surrounding group is missing the output is ignored because of `\aftergroup`. Otherwise output is called some time later when the surrounding group closes. But `\mybox` is void outside the execution phase of the redefined `\shipout`. Also `\@output` checks for a void box and cancels the page output. The disadvantage remains that the hook in `\@output` is called for a page that will not be output.

2.3.3 `\lastkern` method

At the beginning of a new box, there is no `\kern`, the contents of the box is still empty and `\lastkern` returns 0 pt. This can be used to distinguish between direct and indirect boxes: We execute `\setbox` in a box with a preceding non-zero kern. After an indirect box, `\lastkern` sees this kern, otherwise it returns 0 pt.

```
\shipout :=
\begingroup
\setbox\mybox=\hbox\bgroup
\kern1pt
\afterassignment\shipout@test
\global\setbox\mybox=

\@test :=
\ifdim\lastkern=0pt
% direct box
\aftergroup\egroup
\aftergroup\endgroup
\aftergroup\@output
\else
\egroup
\endgroup
```

```

\@output
\fi

```

We have two `\setbox` commands. The first creates a controlled context box where we can safely insert a `\kern`. We get rid of this temporarily used context box by putting the local `\setbox` in a group.

After the group we want to have our shipout box in `\mybox`. Therefore we use a global assignment here.

2.4 Output

With or without ε -TeX we ensure the original behaviour of `\shipout` that void boxes do not generate output pages.

Now we can place the hook `\@hook` for the user code that wants to manipulate the output box.

```

\@output :=
\ifvoid\mybox
% cancel output of void box
\else
\@hook
\ifvoid\mybox
% user code in \@hook could has voided the box
\else
\original@shipout\box\mybox
\fi
\fi

```

2.5 Separate box register

So far we have said nothing about the box number of `\mybox`. The following case that outputs the same page twice shows that we are not free in the use of the box register:

```

\shipout\copy<num> \shipout\box<num>

```

We manipulate the box by the hook and without ε -TeX the box must even be voided. However, the use case above requires that the box contents does not change at all. Therefore we must reserve a separate box register to avoid collisions with user box registers.

Note: Box register number 255 is special for the output routine, because TeX complains if this box is not voided by the output routine. However, this requirement does not apply to `\shipout` at all. In fact `\shipout` does not change any box register. This is usually done by a call of `\box`, but the output routine can do it later *after* invoking of `\shipout`.

2.6 Summary

2.6.1 With ε -TeX

Putting the pieces together we get for ε -TeX:

```

\newbox\mybox
\let\original@shipout\shipout

\shipout :=
\edef\saved@grouplevel{\number\currentgrouplevel}
\afterassignment\@test
\setbox\mybox=

\@test :=
\ifnum\saved@grouplevel<\currentgrouplevel

```



```

\expandafter\aftergroup
\fi
\@output

\@output :=
\ifvoid\mybox
% cancel output of void box
\else
\@hook
\ifvoid\mybox
% user code in \@hook could have voided the box
\else
\original@shipout\box\mybox
\fi
\fi

```

2.6.2 Without ε - \TeX , traditional way

And for \TeX without ε - \TeX :

```

\newbox\mybox
\begingroup
\setbox\mybox=\box\mybox % ensure \mybox is void
\endgroup
\let\original@shipout\shipout

\shipout :=
% trick to get a void box \mybox
\begingroup
\setbox\mybox=\box\mybox
\endgroup
\afterassignment\@test
\setbox\mybox=

\@test :=
\ifvoid\mybox
\expandafter\aftergroup
\fi
\@output

\@output :=
\ifvoid\mybox
% cancel output of void box
\else
\@hook
\ifvoid\mybox
% user code in \@hook could have voided the box
\else
\original@shipout\box\mybox
\fi
\fi

```

2.6.3 $\backslash\text{lastkern}$ method

And for \TeX without ε - \TeX using the $\backslash\text{lastkern}$ method:

```

\newbox\mybox
\let\original@shipout\shipout

\shipout :=
\begingroup
\setbox\mybox=\hbox\bgroup
\kern1pt

```

```

\afterassignment\@test
\setbox\mybox=

\@test :=
\ifdim\lastkern=0pt
\expandafter\aftergroup
\fi
\@output

\@output :=
\egroup
\endgroup
\ifvoid\mybox
% cancel output of void box
\else
\@hook
\ifvoid\mybox
% user code in \@hook could have voided the box
\else
\original@shipout\box\mybox
\fi
\fi

```

3 Implementation

Package `atbegshi` uses ε -TeX's `\currentgrouplevel`, if it is available. Otherwise the `\lastkern` method is used.

```
56 \*package\
```

3.1 Reload check and package identification

Reload check, especially if the package is not used with L^AT_EX.

```

57 \begingroup
58 \catcode44 12 % ,
59 \catcode45 12 % -
60 \catcode46 12 % .
61 \catcode58 12 % :
62 \catcode64 11 % @
63 \catcode123 1 % {
64 \catcode125 2 % }
65 \expandafter\let\expandafter\x\csname ver@atbegshi.sty\endcsname
66 \ifx\x\relax % plain-TeX, first loading
67 \else
68 \def\empty{}%
69 \ifx\x\empty % LaTeX, first loading,
70 % variable is initialized, but \ProvidesPackage not yet seen
71 \else
72 \catcode35 6 % #
73 \expandafter\ifx\csname PackageInfo\endcsname\relax
74 \def\x#1#2{%
75 \immediate\write-1{Package #1 Info: #2.}%
76 }%
77 \else
78 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
79 \fi
80 \x{atbegshi}{The package is already loaded}%
81 \aftergroup\endinput
82 \fi
83 \fi
84 \endgroup

```

Package identification:

```

85 \begingroup
86   \catcode35 6 % #
87   \catcode40 12 % (
88   \catcode41 12 % )
89   \catcode44 12 % ,
90   \catcode45 12 % -
91   \catcode46 12 % .
92   \catcode47 12 % /
93   \catcode58 12 % :
94   \catcode64 11 % @
95   \catcode91 12 % [
96   \catcode93 12 % ]
97   \catcode123 1 % {
98   \catcode125 2 % }
99   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
100     \def\x#1#2#3[#4]{\endgroup
101       \immediate\write-1{Package: #3 #4}%
102       \xdef#1{#4}%
103     }%
104   \else
105     \def\x#1#2[#3]{\endgroup
106       #2[#{#3}]%
107       \ifx#1@undefined
108         \xdef#1{#3}%
109       \fi
110       \ifx#1\relax
111         \xdef#1{#3}%
112       \fi
113     }%
114   \fi
115 \expandafter\x\csname ver@atbegshi.sty\endcsname
116 \ProvidesPackage{atbegshi}%
117 [2010/03/01 v1.11 At begin shipout hook (H0)]

```

3.2 Catcodes

```

118 \begingroup
119   \catcode123 1 % {
120   \catcode125 2 % }
121   \def\x{\endgroup
122     \expandafter\edef\csname AtBegShi@AtEnd\endcsname{%
123       \catcode35 \the\catcode35\relax
124       \catcode64 \the\catcode64\relax
125       \catcode123 \the\catcode123\relax
126       \catcode125 \the\catcode125\relax
127     }%
128   }%
129 \x
130 \catcode35 6 % #
131 \catcode64 11 % @
132 \catcode123 1 % {
133 \catcode125 2 % }
134 \def\TMP@EnsureCode#1#2{%
135   \edef\AtBegShi@AtEnd{%
136     \AtBegShi@AtEnd
137     \catcode#1 \the\catcode#1\relax
138   }%
139   \catcode#1 #2\relax
140 }
141 \TMP@EnsureCode{40}{12}% (
142 \TMP@EnsureCode{41}{12}% )

```

```

143 \TMP@EnsureCode{44}{12}% ,
144 \TMP@EnsureCode{45}{12}% -
145 \TMP@EnsureCode{47}{12}% /
146 \TMP@EnsureCode{46}{12}% .
147 \TMP@EnsureCode{58}{12}% :
148 \TMP@EnsureCode{61}{12}% =
149 \TMP@EnsureCode{94}{7}% ^ (superscript)
150 \TMP@EnsureCode{96}{12}% ‘

```

3.3 Preparations

```

151 \begingroup\expandafter\expandafter\expandafter\endgroup
152 \expandafter\ifx\csname RequirePackage\endcsname\relax
153   \input infwarerr.sty\relax
154   \input ltxcmds.sty\relax
155 \else
156   \RequirePackage{infwarerr}[2007/09/09]%
157   \RequirePackage{ltxcmds}[2010/03/01]%
158 \fi

```

\AtBegShi@CheckDefinable

```

159 \begingroup\expandafter\expandafter\expandafter\endgroup
160 \expandafter\ifx\csname @ifdefinable\endcsname\relax
161   \def\AtBegShi@CheckDefinable#1{%
162     \ifcase\ifx#1\relax
163       \ltx@one
164     \else
165       \ifx#1\@undefined
166         \ltx@one
167       \else
168         \ltx@zero
169       \fi
170     \fi
171     \errmessage{%
172       Package atbegshi: \string#1\space
173       is already defined%
174     }%
175   \endgroup
176 \fi
177 }%
178 \else
179   \def\AtBegShi@CheckDefinable#1{%
180     \@ifdefinable{#1}{}%
181   }%
182 \fi

183 \ltx@newif\ifAtBegShi@Discarded

```

\AtBeginShipoutDiscard

```

184 \AtBegShi@CheckDefinable\AtBeginShipoutDiscard
185 \def\AtBeginShipoutDiscard{%
186   \deadcycles=\ltx@zero
187   \global\AtBegShi@Discardedtrue
188 }

189 \begingroup\expandafter\expandafter\expandafter\endgroup
190 \expandafter\ifx\csname currentgrouplevel\endcsname\relax
191   \catcode‘X=9 % ignore
192   \catcode‘E=14 % comment
193 \else
194   \catcode‘X=14 % comment
195   \catcode‘E=9 % ignore
196 \fi

```

\AtBegShi@Shipout

```
197 \def\AtBegShi@Shipout{%
198 X \begingroup
199 X \setbox\AtBeginShipoutBox=\hbox\bgroup
200 X \kern\p@
201 E \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
202 \afterassignment\AtBegShi@Test
203 X \global
204 \setbox\AtBeginShipoutBox=%
205 }
```

\AtBegShi@Test

```
206 \def\AtBegShi@Test{%
207 X \ifdim\lastkern=0pt %
208 E \ifnum\AtBegShi@GroupLevel<\currentgrouplevel
209 \expandafter\aftergroup
210 \fi
211 \AtBegShi@Output
212 }
```

\AtBegShi@Output

```
213 \def\AtBegShi@Output{%
214 X \egroup
215 X \endgroup
216 \ifvoid\AtBeginShipoutBox
217 \PackageWarning{atbegshi}{Ignoring void shipout box}%
218 \else
219 \let\AtBegShi@OrgProtect\protect
220 \csname set@typeset@protect\endcsname
221 \global\AtBegShi@Discardedfalse
222 \AtBegShi@Hook
223 \AtBegShi@HookNext
224 \gdef\AtBegShi@HookNext{}%
225 \ifAtBegShi@Discarded
226 \PackageInfoNoLine{atbegshi}{Shipout page discarded}%
227 \global\AtBegShi@Discardedfalse
228 \begingroup
229 \setbox\AtBeginShipoutBox\box\AtBeginShipoutBox
230 \endgroup
231 \let\protect\AtBegShi@OrgProtect
232 \else
233 \AtBegShi@First
234 \let\protect\AtBegShi@OrgProtect
235 \AtBeginShipoutOriginalShipout\box\AtBeginShipoutBox
236 \fi
237 \fi
238 }

239 \catcode'\X=11 %
240 \catcode'\E=11 %
```

\AtBegShi@First

```
241 \def\AtBegShi@First{%
242 \begingroup
243 \def\@empty{}%
244 \ifx\AtBegShi@HookFirst\@empty
245 \else
246 \setbox\ltx@zero=\vbox{%
247 \begingroup
248 \AtBegShi@HookFirst
249 \endgroup
250 }%
```

```

251      \wd\ltx@zero=Opt %
252      \ht\ltx@zero=Opt %
253      \dp\ltx@zero=Opt %
254      \global\setbox\AtBeginShipoutBox=\vbox{%
255        \baselineskip Opt\relax
256        \lineskip Opt\relax
257        \lineskiplimit Opt\relax
258        \copy\ltx@zero
259        \copy\AtBeginShipoutBox
260      }%
261      \fi
262      \global\let\AtBegShi@First\@empty
263      \global\let\AtBeginShipoutFirst\AtBegShi@FirstDisabled
264    \endgroup
265  }

\AtBegShi@Hook
266 \gdef\AtBegShi@Hook{}

\AtBegShi@HookNext
267 \gdef\AtBegShi@HookNext{}

\AtBegShi@HookFirst
268 \gdef\AtBegShi@HookFirst{}

\AtBeginShipout
269 \AtBegShi@CheckDefinable\AtBeginShipout
270 \def\AtBeginShipout{%
271   \AtBegShi@AddHook\AtBegShi@Hook
272 }

\AtBeginShipoutNext
273 \AtBegShi@CheckDefinable\AtBeginShipoutNext
274 \def\AtBeginShipoutNext{%
275   \AtBegShi@AddHook\AtBegShi@HookNext
276 }

\AtBeginShipoutFirst
277 \AtBegShi@CheckDefinable\AtBeginShipoutFirst
278 \def\AtBeginShipoutFirst{%
279   \AtBegShi@AddTo\AtBegShi@HookFirst
280 }

\AtBegShi@FirstDisabled
281 \long\def\AtBegShi@FirstDisabled#1{%
282   \@PackageWarning{atbegshi}{%
283     First page is already shipped out, ignoring\MessageBreak
284     \string\AtBeginShipoutFirst
285   }%
286 }

\AtBegShi@AddTo
287 \begingroup\expandafter\expandafter\expandafter\endgroup
288 \expandafter\ifx\csname g@addto@macro\endcsname\relax
289   \long\def\AtBegShi@AddTo#1#2{%
290     \begingroup
291       \toks\ltx@zero\expandafter{#1#2}%
292       \xdef#1{\the\toks\ltx@zero}%
293     \endgroup
294   }%
295 \else

```

```

296 \let\AtBegShi@AddTo\g@addto@macro
297 \fi

\AtBegShi@AddHook

298 \long\def\AtBegShi@AddHook#1#2{%
299 \AtBegShi@AddTo#1{\AtBegShi@Item{#2}}%
300 }

\AtBegShi@Item

301 \long\def\AtBegShi@Item#1{%
302 \ifAtBegShi@Discarded
303 \else
304 #1%
305 \ifvoid\AtBeginShipoutBox
306 \@PackageWarning{atbegshi}{%
307 Shipout box was voided by hook,\MessageBreak
308 ignoring shipout box%
309 }%
310 \AtBeginShipoutDiscard
311 \fi
312 \fi
313 }

\AtBeginShipoutInit

314 \AtBegShi@CheckDefinable\AtBeginShipoutInit
315 \def\AtBeginShipoutInit{%
316 \ltx@ifundefined{newbox}{%
317 \@PackageError{atbegshi}{%
318 \string\AtBeginShipoutInit\space failed\MessageBreak
319 because of missing \expandafter\string\csname newbox\endcsname
320 }\@ehc
321 }{%
322 \csname newbox\endcsname\AtBeginShipoutBox
323 \AtBegShi@CheckDefinable\AtBeginShipoutOriginalShipout
324 \global\let\AtBeginShipoutOriginalShipout\shipout
325 \global\let\shipout\AtBegShi@Shipout
326 }%
327 \gdef\AtBeginShipoutInit{}%
328 }

329 \begingroup\expandafter\expandafter\expandafter\endgroup
330 \expandafter\ifx\csname AtBeginDocument\endcsname\relax
331 \AtBeginShipoutInit
332 \else
333 \AtBeginDocument{\AtBeginShipoutInit}%
334 \fi

```

3.4 Positioning

```

335 \begingroup\expandafter\expandafter\expandafter\endgroup
336 \expandafter\ifx\csname RequirePackage\endcsname\relax
337 \input ifpdf.sty\relax
338 \else
339 \RequirePackage{ifpdf}\relax
340 \fi

341 \ifpdf
342 \def\AtBegShi@horigin{\pdfhorigin}%
343 \def\AtBegShi@vorigin{\pdfvorigin}%
344 \else
345 \def\AtBegShi@horigin{72.27pt}%
346 \def\AtBegShi@vorigin{72.27pt}%
347 \fi

```

```

348 \begingroup
349 \ifcase
350   \expandafter\ifx\csname picture\endcsname\relax
351     1%
352   \else
353     \expandafter\ifx\csname endpicture\endcsname\relax
354       1%
355     \else
356       0%
357     \fi
358   \fi
359 \endgroup
360 \def\AtBegShi@BeginPicture{%
361   \begingroup
362   \picture(0,0)\relax
363   \begingroup\expandafter\expandafter\expandafter\endgroup
364   \expandafter\ifx\csname unitlength\endcsname\relax
365   \else
366     \unitlength=1pt\relax
367   \fi
368   \ignorespaces
369 }%
370 \def\AtBegShi@EndPicture{%
371   \endpicture
372   \endgroup
373 }%
374 \else
375   \endgroup
376   \def\AtBegShi@BeginPicture{%
377     \setbox\ltx@zero=\hbox\bgroup
378     \begingroup
379     \ignorespaces
380   }%
381   \def\AtBegShi@EndPicture{%
382     \endgroup
383     \egroup
384     \ht\ltx@zero=0pt\relax
385     \dp\ltx@zero=0pt\relax
386     \copy\ltx@zero
387   }%
388 \fi
389 \def\AtBeginShipoutUpperLeft#1{%
390   \global\setbox\AtBeginShipoutBox=\hbox{%
391     \rlap{%
392       \kern-\AtBegShi@horigin\relax
393       \vbox to 0pt{%
394         \kern-\AtBegShi@vorigin\relax
395         \kern-\ht\AtBeginShipoutBox
396         \AtBegShi@BeginPicture
397         #1%
398         \AtBegShi@EndPicture
399         \vss
400       }%
401     }%
402     \box\AtBeginShipoutBox
403   }%
404 }
405 \def\AtBeginShipoutUpperLeftForeground#1{%
406   \global\setbox\AtBeginShipoutBox=\hbox to \wd\AtBeginShipoutBox{%
407     \rlap{%
408       \copy\AtBeginShipoutBox
409     }%

```



```

410 \rlap{%
411 \kern-\AtBegShi@horigin\relax
412 \vbox to Opt{%
413 \kern-\AtBegShi@vorigin\relax
414 \kern-\ht\AtBeginShipoutBox
415 \AtBegShi@BeginPicture
416 #1%
417 \AtBegShi@endPicture
418 \vss
419 }%
420 }%
421 \hss
422 }%
423 }

```

3.5 Patches

Patches for L^AT_EX packages that redefine `\shipout`. L^AT_EX is now supposed to use ε -T_EX. Thus we do not patch, without L^AT_EX and ε -T_EX.

```

424 \def\AtBegShi@AbortIfUndefined#1{%
425 \begingroup\expandafter\expandafter\expandafter\endgroup
426 \expandafter\ifx\csname#1\endcsname\relax
427 \AtBegShi@AtEnd
428 \expandafter\endinput
429 \fi
430 }
431 \AtBegShi@AbortIfUndefined{currentgrouplevel}
432 \AtBegShi@AbortIfUndefined{AtBeginDocument}
433 \AtBegShi@AbortIfUndefined{@ifpackageloaded}
434 \AtBegShi@AbortIfUndefined{@ifclassloaded}

```

3.5.1 Package crop

Fix of method and box.

```

435 \def\AtBegShi@PatchCrop{%
436 \begingroup
437 \def\AtBegShi@Crop@shipout{%
438 \afterassignment\CROP@ship
439 \setbox\@cclv=%
440 }%
441 \def\AtBegShi@Crop@ship{%
442 \ifvoid\@cclv
443 \expandafter\aftergroup
444 \fi
445 \CROP@@ship
446 }%
447 \def\AtBegShi@Crop@shiplist{%
448 \lineskip\z@
449 \lineskiplimit\z@
450 \baselineskip\z@
451 \CROP@kernel
452 \box\@cclv
453 }%
454 \def\AtBegShi@Crop@@ship{%
455 \CROP@shipout\vbox{%
456 \CROP@shiplist
457 }%
458 }%
459 \ifx\AtBegShi@Crop@ship\CROP@ship
460 \ifx\AtBegShi@Crop@shiplist\CROP@shiplist
461 \ifx\AtBegShi@Crop@@ship\CROP@@ship
462 \let\AtBegShi@found\relax
463 \ifx\shipout\AtBegShi@Crop@shipout

```

```

464         \def\AtBegShi@found{\shipout}%
465     \else\ifx\AtBeginShipoutOriginalShipout\AtBegShi@Crop@shipout
466         \def\AtBegShi@found{\AtBeginShipoutOriginalShipout}%
467     \else\ifx\@EveryShipout@Org@Shipout\AtBegShi@Crop@shipout
468         \def\AtBegShi@found{\@EveryShipout@Org@Shipout}%
469     \else\ifx\GPTorg@shipout\AtBegShi@Crop@shipout
470         \def\AtBegShi@found{\GPTorg@shipout}%
471     \else\ifx\THBorg@shipout\AtBegShi@Crop@shipout
472         \def\AtBegShi@found{\THBorg@shipout}%
473     \else\ifx\mem@oldshipout\AtBegShi@Crop@shipout
474         \def\AtBegShi@found{\mem@oldshipout}%
475     \fi\fi\fi\fi\fi\fi
476     \ifx\AtBegShi@found\relax
477     \else
478         \expandafter\endgroup
479         \expandafter\def\AtBegShi@found{%
480             \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
481             \afterassignment\CROP@ship
482             \setbox\AtBeginShipoutBox=%
483         }%
484         \def\CROP@ship{%
485             \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
486             \else
487                 \expandafter\aftergroup
488                 \fi
489             \CROP@@ship
490         }%
491         \def\CROP@shiplist{%
492             \lineskip Opt\relax
493             \lineskiplimit Opt\relax
494             \baselineskip Opt\relax
495             \CROP@kernel
496             \box\AtBeginShipoutBox
497         }%
498         \def\CROP@@ship{%
499             \ifvoid\AtBeginShipoutBox
500             \else
501                 \setbox\AtBeginShipoutBox=\vbox{%
502                     \CROP@shiplist
503                 }%
504                 \expandafter\CROP@shipout
505                 \expandafter\box
506                 \expandafter\AtBeginShipoutBox
507             \fi
508         }%
509         \@PackageInfoNoLine{atbegshi}{Package ‘crop’ patched}%
510         \begingroup
511         \fi
512     \fi
513 \fi
514 \fi
515 \endgroup
516 \let\AtBegShi@PatchCrop\relax
517 }
518 \ifpackageloaded{crop}{%
519     \AtBegShi@PatchCrop
520 }{%
521     \AtBeginDocument{\AtBegShi@PatchCrop}%
522 }

```

3.5.2 Package everyshi

Fix of method. Use of box 255 is not changed.

```
523 \def\AtBegShi@PatchEveryshi{%
524   \begingroup
525     \long\def\AtBegShi@Everyshi@shipout{%
526       \afterassignment\@EveryShipout@Test
527       \global\setbox\@cclv= %
528     }%
529   \long\def\AtBegShi@Everyshi@Test{%
530     \ifvoid\@cclv\relax
531       \aftergroup\@EveryShipout@Output
532     \else
533       \@EveryShipout@Output
534     \fi
535   }%
536   \ifx\AtBegShi@Everyshi@Test\@EveryShipout@Test
537     \let\AtBegShi@found\relax
538     \ifx\shipout\AtBegShi@Everyshi@shipout
539       \def\AtBegShi@found{\shipout}%
540     \else\ifx\AtBeginShipoutOriginalShipout\AtBegShi@Everyshi@shipout
541       \def\AtBegShi@found{\AtBeginShipoutOriginalShipout}%
542     \else\ifx\CROP@shipout\AtBegShi@Everyshi@shipout
543       \def\AtBegShi@found{\CROP@shipout}%
544     \else\ifx\GPTorg@shipout\AtBegShi@Everyshi@shipout
545       \def\AtBegShi@found{\GPTorg@shipout}%
546     \else\ifx\THBorg@shipout\AtBegShi@Everyshi@shipout
547       \def\AtBegShi@found{\THBorg@shipout}%
548     \else\ifx\mem@oldshipout\AtBegShi@Everyshi@shipout
549       \def\AtBegShi@found{\mem@oldshipout}%
550     \else
551       \expandafter\ifx\csname @EveryShipout@Org@Shipout\endcsname
552         \relax
553         \ifx\@EveryShipout@Shipout\AtBegShi@Everyshi@shipout
554           \def\AtBegShi@found{\@EveryShipout@Shipout}%
555         \fi
556       \fi
557     \fi\fi\fi\fi\fi\fi
558     \ifx\AtBegShi@found\relax
559     \else
560       \expandafter\endgroup
561       \expandafter\def\AtBegShi@found{%
562         \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
563         \afterassignment\@EveryShipout@Test
564         \setbox\AtBeginShipoutBox=%
565       }%
566       \def\@EveryShipout@Test{%
567         \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
568         \else
569           \expandafter\aftergroup
570         \fi
571         \AtBegShi@Everyshi@Output
572       }%
573       \def\AtBegShi@Everyshi@Output{%
574         \ifvoid\AtBeginShipoutBox
575         \else
576           \global\setbox\ltx@cclv\box\AtBeginShipoutBox
577           \expandafter\@EveryShipout@Output
578         \fi
579       }%
580       \@PackageInfoNoLine{atbegshi}{Package 'everyshi' patched}%
581     \begingroup
582   \fi
```

```

583 \fi
584 \endgroup
585 \let\AtBegShi@PatchEveryshi\relax
586 }
587 \@ifpackageloaded{everyshi}{%
588 \AtBegShi@PatchEveryshi
589 }{%
590 \AtBeginDocument{\AtBegShi@PatchEveryshi}%
591 }

```

3.5.3 Class memoir

Fix of method and box.

```

592 \def\AtBegShi@PatchMemoir{%
593 \begingroup
594 \def\AtBegShi@Memoir@shipout{%
595 \afterassignment\mem@shipi
596 \setbox\@cclv=%
597 }%
598 \def\AtBegShi@Memoir@shipi{%
599 \ifvoid\@cclv
600 \expandafter\aftergroup
601 \fi
602 \mem@shipii
603 }%
604 \def\AtBegShi@Memoir@shipiiA{%
605 \mem@oldshipout\ vbox{%
606 \trimmarks
607 \unvbox\@cclv
608 }%
609 }%
610 \def\AtBegShi@Memoir@shipiiB{%
611 \ifvoid\@cclv
612 \mem@oldshipout\ box\@cclv
613 \else
614 \mem@oldshipout\ vbox{%
615 \trimmarks
616 \unvbox\@cclv
617 }%
618 \fi
619 }%
620 \ifx\AtBegShi@Memoir@shipi\mem@shipi
621 \ifcase\ifx\AtBegShi@Memoir@shipiiA\mem@shipii
622 \ltx@zero
623 \else
624 \ifx\AtBegShi@Memoir@shipiiB\mem@shipii
625 \ltx@zero
626 \else
627 \ltx@one
628 \fi
629 \fi
630 \let\AtBegShi@found\relax
631 \ifx\shipout\AtBegShi@Memoir@shipout
632 \def\AtBegShi@found{\shipout}%
633 \else\ifx\AtBeginShipoutOriginalShipout\AtBegShi@Memoir@shipout
634 \def\AtBegShi@found{\AtBeginShipoutOriginalShipout}%
635 \else\ifx\CROP@shipout\AtBegShi@Memoir@shipout
636 \def\AtBegShi@found{\CROP@shipout}%
637 \else\ifx\GPTorg@shipout\AtBegShi@Memoir@shipout
638 \def\AtBegShi@found{\GPTorg@shipout}%
639 \else\ifx\THBorg@shipout\AtBegShi@Memoir@shipout
640 \def\AtBegShi@found{\THBorg@shipout}%

```

```

641 \else\ifx\@EveryShipout@Org@Shipout\AtBegShi@Memoir@shipout
642 \def\AtBegShi@found{\@EveryShipout@Org@Shipout}%
643 \fi\fi\fi\fi\fi\fi
644 \ifx\AtBegShi@found\relax
645 \else
646 \expandafter\endgroup
647 \expandafter\def\AtBegShi@found{%
648 \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
649 \afterassignment\mem@shipi
650 \setbox\AtBeginShipoutBox=%
651 }%
652 \def\mem@shipi{%
653 \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
654 \else
655 \expandafter\aftergroup
656 \fi
657 \mem@shipii
658 }%
659 \def\mem@shipii{%
660 \ifvoid\AtBeginShipoutBox
661 \else
662 \setbox\AtBeginShipoutBox=\vbox{%
663 \trimmarks
664 \ifvbox\AtBeginShipoutBox
665 \unvbox\AtBeginShipoutBox
666 \else
667 \box\AtBeginShipoutBox
668 \fi
669 }%
670 \expandafter\mem@oldshipout
671 \expandafter\box
672 \expandafter\AtBeginShipoutBox
673 \fi
674 }%
675 \@PackageInfoNoLine{atbegshi}{Class 'memoir' patched}%
676 \begingroup
677 \fi
678 \fi
679 \fi
680 \endgroup
681 \let\AtBegShi@PatchMemoir\relax
682 }
683 \@ifclassloaded{memoir}{%
684 \AtBegShi@PatchMemoir
685 }{%
686 \AtBeginDocument{\AtBegShi@PatchMemoir}%
687 }
688 \AtBegShi@AtEnd
689 \end{package}

```

4 Test

4.1 Catcode checks for loading

```

690 \test1
691 \catcode'\{=1 %
692 \catcode'\}=2 %
693 \catcode'\#=6 %
694 \catcode'\@=11 %
695 \expandafter\ifx\csname count@\endcsname\relax
696 \countdef\count@=255 %

```

```

697 \fi
698 \expandafter\ifx\csname @gobble\endcsname\relax
699 \long\def@gobble#1{%
700 \fi
701 \expandafter\ifx\csname @firstofone\endcsname\relax
702 \long\def@firstofone#1{#1}%
703 \fi
704 \expandafter\ifx\csname loop\endcsname\relax
705 \expandafter@firstofone
706 \else
707 \expandafter@gobble
708 \fi
709 {%
710 \def\loop#1\repeat{%
711 \def\body{#1}%
712 \iterate
713 }%
714 \def\iterate{%
715 \body
716 \let\next\iterate
717 \else
718 \let\next\relax
719 \fi
720 \next
721 }%
722 \let\repeat=\fi
723 }%
724 \def\RestoreCatcodes{}
725 \count@=0 %
726 \loop
727 \edef\RestoreCatcodes{%
728 \RestoreCatcodes
729 \catcode\the\count@=\the\catcode\count@\relax
730 }%
731 \ifnum\count@<255 %
732 \advance\count@ 1 %
733 \repeat
734
735 \def\RangeCatcodeInvalid#1#2{%
736 \count@=#1\relax
737 \loop
738 \catcode\count@=15 %
739 \ifnum\count@<#2\relax
740 \advance\count@ 1 %
741 \repeat
742 }
743 \expandafter\ifx\csname LoadCommand\endcsname\relax
744 \def\LoadCommand{\input atbegshi.sty\relax}%
745 \fi
746 \def\Test{%
747 \RangeCatcodeInvalid{0}{47}%
748 \RangeCatcodeInvalid{58}{64}%
749 \RangeCatcodeInvalid{91}{96}%
750 \RangeCatcodeInvalid{123}{255}%
751 \catcode'\@=12 %
752 \catcode'\=0 %
753 \catcode'\{=1 %
754 \catcode'\}=2 %
755 \catcode'\#=6 %
756 \catcode'\[=12 %
757 \catcode'\]=12 %
758 \catcode'\%=14 %

```

```

759 \catcode'\ =10 %
760 \catcode13=5 %
761 \LoadCommand
762 \RestoreCatcodes
763 }
764 \Test
765 \csname @@end\endcsname
766 \end

767 </test1>

768 <*test2>
769 \input atbegshi.sty\relax
770 \def\msg#\{\immediate\write16}
771 \msg{File: atbegshi-test2.tex 2010/03/01 v1.11 Test file for plain-TeX}
772 \def\testmsg#1#2{%
773   \msg{}%
774   \msg{*** Test with box (#1), expected page output [#2]}}% hash-ok
775 }
776
777 \newbox\voidbox
778 \def\void{\box\voidbox}
779 \begingroup
780   \setbox\voidbox=\void
781 \endgroup
782
783 \count0=0\relax
784 \AtBeginShipout{%
785   \global\advance\count0 by 1\relax
786   \msg{* Inside \string\AtBeginShipout: [\the\count0]}}%
787 }
788
789 \AtBeginShipoutFirst{%
790   \msg{* Inside \string\AtBeginShipoutFirst}%
791   Hello World%
792 }
793
794 \testmsg{\string\null}{1}
795 \shipout\null
796
797 \AtBeginShipoutFirst{%
798   This is too late%
799 }
800
801 \testmsg{void}{}
802 \shipout\void
803
804 \testmsg{\string\copy255 (not void)}{2}
805 \setbox255\hbox{\vrule height 10bp width 10bp}
806 \shipout\copy255 %
807
808 \testmsg{\string\copy255 (again)}{3}
809 \shipout\copy255 %
810
811 \testmsg{\string\box255}{4}
812 \shipout\box255 %
813
814 \testmsg{\string\box255 (again)}{}
815 \shipout\box255 %
816
817 \testmsg{\string\hbox}{5}
818 \shipout\hbox{\vrule height 5bp width 20bp}
819
820 \testmsg{\string\vbox}{6}

```

```

821 \shipout\vbox{\hrule height 20bp width 5bp}
822
823 \testmsg{\string\null, voided by hook}{%
824 \def\VoidBox{%
825   \begingroup
826     \setbox\AtBeginShipoutBox=\box\AtBeginShipoutBox
827   \endgroup
828 }
829 \AtBeginShipout{\VoidBox}
830 \shipout\null
831 \def\VoidBox{}
832
833 \msg{*** \string\begingroup}
834 \begingroup
835   \testmsg{void}{}%
836   \shipout\void
837 \msg{*** \string\endgroup}
838 \endgroup
839
840 \msg{*** \string\begingroup}
841 \begingroup
842   \testmsg{void}{}%
843   \shipout\void
844   \testmsg{\string\null}{8}%
845   \shipout\null
846 \msg{*** \string\endgroup}
847 \endgroup
848
849 \testmsg{output routine}{9}
850 Hello World
851 \vfill
852 \eject
853
854 \testmsg{\string\null\space(discarded)}{0}
855 \AtBeginShipout{%
856   \msg{* Inside \string\AtBeginShipout: DISCARD}%
857   \AtBeginShipoutDiscard
858 }
859 \shipout\null
860
861 \end
862 </test2>
863 <*test3>
864 \NeedsTeXFormat{LaTeX2e}
865 \ProvidesFile{atbegshi-test3.tex}[2010/03/01 v1.11 Test file for LaTeX]
866 \RequirePackage{color}
867 \pagecolor{yellow}
868 \documentclass[a5paper,showtrims]{memoir}
869 \usepackage{atbegshi}
870 \AtBeginShipout{%
871   \setbox\AtBeginShipoutBox=\vbox{%
872     \vbox to 0pt{%
873       \kern-1.5in %
874       \hbox to 0pt{%
875         \kern-1.5in %
876         \color{blue}%
877         \rule{1in}{1in}%
878       \hss
879     }%
880     \vss
881   }%
882   \hrule

```



```

883     \hbox{\vrule\box\AtBeginShipoutBox\vrule}%
884     \hrule
885   }%
886 }
887 \usepackage{eso-pic}
888 \makeatletter
889 \@ifundefined{EveryShipout@Init}{%
890   \typeout{Test skipped}%
891   \@end
892 }{}
893 \@EveryShipout@Init
894 \let\EveryShipout@Init\relax
895 \makeatother
896 \AddToShipoutPicture{%
897   \hspace{.52\paperwidth}%
898   \colorbox{cyan}{%
899     \rule{0mm}{\paperheight}%
900     \hspace{.48\paperwidth}%
901   }%
902 }

```

Newer versions of class memoir emulate package crop and prevents its loading. This is undone in next line for this test file.

```

903 \expandafter\let\csname ver@crop.sty\endcsname\relax
904 \usepackage[color=red,cross,a4,center]{crop}
905 \begin{document}
906 \shipout\null
907 \shipout\box\csname voidb@x\endcsname
908 \section{Hello World}
909 \end{document}
910 </test3>

```

5 Installation

5.1 Download

Package. This package is available on CTAN¹:

[CTAN:macros/latex/contrib/oberdiek/atbegshi.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/atbegshi.pdf](#) Documentation.

Bundle. All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](#)

TDS refers to the standard “A Directory Structure for \TeX Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

5.2 Bundle installation

Unpacking. Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

¹<http://ftp.ctan.org/tex-archive/>

Script installation. Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

5.3 Package installation

Unpacking. The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain-`TEX`:

```
tex atbegshi.dtx
```

TDS. Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

<code>atbegshi.sty</code>	→ <code>tex/generic/oberdiek/atbegshi.sty</code>
<code>atbegshi.pdf</code>	→ <code>doc/latex/oberdiek/atbegshi.pdf</code>
<code>atbegshi-example1.tex</code>	→ <code>doc/latex/oberdiek/atbegshi-example1.tex</code>
<code>atbegshi-example2.tex</code>	→ <code>doc/latex/oberdiek/atbegshi-example2.tex</code>
<code>test/atbegshi-test1.tex</code>	→ <code>doc/latex/oberdiek/test/atbegshi-test1.tex</code>
<code>test/atbegshi-test2.tex</code>	→ <code>doc/latex/oberdiek/test/atbegshi-test2.tex</code>
<code>test/atbegshi-test3.tex</code>	→ <code>doc/latex/oberdiek/test/atbegshi-test3.tex</code>
<code>atbegshi.dtx</code>	→ <code>source/latex/oberdiek/atbegshi.dtx</code>

If you have a `docstrip.cfg` that configures and enables `docstrip`'s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

5.4 Refresh file name databases

If your `TEX` distribution (te`TEX`, mik`TEX`, ...) relies on file name databases, you must refresh these. For example, te`TEX` users run `texhash` or `mktextlsr`.

5.5 Some details for the interested

Attached source. The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk atbegshi.pdf unpack_files output .
```

Unpacking with L^AT_EX. The `.dtx` chooses its action depending on the format:

plain-`TEX`: Run `docstrip` and extract the files.

L^AT_EX: Generate the documentation.

If you insist on using L^AT_EX for `docstrip` (really, `docstrip` does not need L^AT_EX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{atbegshi.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfL^AT_EX:

```
pdflatex atbegshi.dtx
makeindex -s gind.ist atbegshi.idx
pdflatex atbegshi.dtx
makeindex -s gind.ist atbegshi.idx
pdflatex atbegshi.dtx
```

6 History

[2007/04/17 v1.0]

- First version.

[2007/04/18 v1.1]

- New method based on `\lastkern` is used if ϵ -T_EX is missing.
- `\AtBeginShipoutDiscard` also resets `\deadcycles`.

[2007/04/19 v1.2]

- `\AtBeginShipoutEarly` removed for simplification reasons.
- Forgotten definition of `\AtBegShi@Info` added.
- Patches for packages `crop` and `everyshi` and class `memoir` added.

[2007/04/26 v1.3]

- Use of package `infwarerr`.
- Catcode section after generic header.

[2007/04/27 v1.4]

- Small optimizations.

[2007/06/06 v1.5]

- `\AtBeginShipoutUpperLeft` added.
- Example added.
- Fix in second test file for newer version of `memoir`.

[2007/09/09 v1.6]

- Catcode section rewritten.

[2008/07/18 v1.7]

- Documentation of `\AtBeginShipoutUpperLeft` fixed and extended.

[2008/07/19 v1.8]

- `\AtBeginShipoutUpperLeftForeground` added.

[2008/07/31 v1.9]

- Second example (`TrimBox` for `dvipdfmx`) added.
- No changes in package code.

[2009/12/02 v1.10]

- `\AtBeginShipoutOriginalShipout` added.
- Test file fixed.

[2010/03/01 v1.11]

- Compatibility with `ini-TeX` except for `\newbox`.

7 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.

Symbols	
<code>\#</code>	693, 755
<code>\%</code>	758
<code>\@</code>	694, 751
<code>\@end</code>	891
<code>\@EveryShipout@Init</code>	893, 894
<code>\@EveryShipout@Org@Shipout</code>	467, 468, 641, 642
<code>\@EveryShipout@Output</code> ..	531, 533, 577
<code>\@EveryShipout@Shipout</code>	553, 554
<code>\@EveryShipout@Test</code> ..	526, 536, 563, 566
<code>\@PackageError</code>	317
<code>\@PackageInfoNoLine</code> ..	226, 509, 580, 675
<code>\@PackageWarning</code>	217, 282, 306
<code>\@ccclv</code>	439, 442, 452, 527, 530, 596, 599, 607, 611, 612, 616
<code>\@ehc</code>	320
<code>\@empty</code>	243, 244, 262
<code>\@firstofone</code>	702, 705
<code>\@gobble</code>	699, 707
<code>\@ifclassloaded</code>	683
<code>\@ifdefinable</code>	180
<code>\@ifpackageloaded</code>	518, 587
<code>\@ifundefined</code>	889
<code>\@undefined</code>	107, 165
<code>\[</code>	756
<code>\]</code>	752
<code>\{</code>	691, 753
<code>\}</code>	692, 754
<code>\]</code>	757
<code>_</code>	759
A	
<code>\AddToShipoutPicture</code>	896
<code>\advance</code>	732, 740, 785
<code>\afterassignment</code>	202, 438, 481, 526, 563, 595, 649
<code>\aftergroup</code>	81, 209, 443, 487, 531, 569, 600, 655
<code>\AtBeginDocument</code> ..	333, 521, 590, 686
<code>\AtBeginShipout</code>	2, 6, 44, 269, 784, 786, 829, 855, 856, 870
<code>\AtBeginShipoutBox</code> ..	45, 47, 199, 204, 216, 229, 235, 254, 259, 305, 322, 390, 395, 402, 406, 408, 414, 482, 496, 499, 501, 506, 564, 574, 576, 650, 660, 662, 664, 665, 667, 672, 826, 871, 883
<code>\AtBeginShipoutDiscard</code>	3, 29, 184, 310, 857
<code>\AtBeginShipoutFirst</code>	3, 263, 277, 284, 789, 790, 797
<code>\AtBeginShipoutInit</code> ..	3, 314, 331, 333
<code>\AtBeginShipoutNext</code> ..	2, 14, 28, 273
<code>\AtBeginShipoutOriginalShipout</code> ..	3, 235, 323, 324, 465, 466, 540, 541, 633, 634
<code>\AtBeginShipoutUpperLeft</code> ..	3, 7, 15, 389
<code>\AtBeginShipoutUpperLeftForeground</code> ..	3, 405
<code>\AtBegShi@AbortIfUndefined</code>	424, 431, 432, 433, 434
<code>\AtBegShi@AddHook</code>	271, 275, 298
<code>\AtBegShi@AddTo</code>	279, 287, 299
<code>\AtBegShi@AtEnd</code> ..	135, 136, 427, 688
<code>\AtBegShi@BeginPicture</code>	360, 376, 396, 415
<code>\AtBegShi@CheckDefinable</code>	159, 184, 269, 273, 277, 314, 323

<code>\AtBegShi@Crop@ship</code>	454, 461	<code>\copy</code>	258,
<code>\AtBegShi@Crop@ship</code>	441, 459		259, 386, 408, 804, 806, 808, 809
<code>\AtBegShi@Crop@shiplist</code> . . .	447, 460	<code>\count</code>	783, 785, 786
<code>\AtBegShi@Crop@shipout</code>		<code>\count@</code>	696, 725,
	437, 463, 465, 467, 469, 471, 473		729, 731, 732, 736, 738, 739, 740
<code>\AtBegShi@Discardedfalse</code> . . .	221, 227	<code>\countdef</code>	696
<code>\AtBegShi@Discardedtrue</code>	187	<code>\CROP@ship</code>	445, 461, 489, 498
<code>\AtBegShi@EndPicture</code> 370, 381, 398, 417		<code>\CROP@kernel</code>	451, 495
<code>\AtBegShi@Everyshi@Output</code> . .	571, 573	<code>\CROP@ship</code>	438, 459, 481, 484
<code>\AtBegShi@Everyshi@shipout</code> .	525,	<code>\CROP@shiplist</code>	456, 460, 491, 502
	538, 540, 542, 544, 546, 548, 553	<code>\CROP@shipout</code>	
<code>\AtBegShi@Everyshi@Test</code> . . .	529, 536		455, 504, 542, 543, 635, 636
<code>\AtBegShi@First</code>	233, 241	<code>\csname</code>	65,
<code>\AtBegShi@FirstDisabled</code> . . .	263, 281		73, 99, 115, 122, 152, 160, 190,
<code>\AtBegShi@found</code> . . .	462, 464, 466,		220, 288, 319, 322, 330, 336,
	468, 470, 472, 474, 476, 479,		350, 353, 364, 426, 551, 695,
	537, 539, 541, 543, 545, 547,		698, 701, 704, 743, 765, 903, 907
	549, 554, 558, 561, 630, 632,	<code>\currentgrouplevel</code>	201,
	634, 636, 638, 640, 642, 644, 647		208, 480, 485, 562, 567, 648, 653
<code>\AtBegShi@GroupLevel</code>	201,		
	208, 480, 485, 562, 567, 648, 653	D	
<code>\AtBegShi@Hook</code>	222, 266, 271	<code>\deadcycles</code>	186
<code>\AtBegShi@HookFirst</code> 244, 248, 268, 279		<code>\documentclass</code>	2, 37, 868
<code>\AtBegShi@HookNext</code> .	223, 224, 267, 275	<code>\dp</code>	253, 385
<code>\AtBegShi@horigin</code> .	342, 345, 392, 411		
<code>\AtBegShi@Item</code>	299, 301	E	
<code>\AtBegShi@Memoir@shipi</code>	598, 620	<code>\E</code>	240
<code>\AtBegShi@Memoir@shipiiiA</code> . . .	604, 621	<code>\eject</code>	852
<code>\AtBegShi@Memoir@shipiiiB</code> . . .	610, 624	<code>\empty</code>	68, 69
<code>\AtBegShi@Memoir@shipout</code>		<code>\end</code>	34, 54, 766, 861, 909
	594, 631, 633, 635, 637, 639, 641	<code>\endcsname</code>	65,
<code>\AtBegShi@OrgProtect</code> . .	219, 231, 234		73, 99, 115, 122, 152, 160, 190,
<code>\AtBegShi@Output</code>	211, 213		220, 288, 319, 322, 330, 336,
<code>\AtBegShi@PatchCrop</code> 435, 516, 519, 521			350, 353, 364, 426, 551, 695,
<code>\AtBegShi@PatchEveryshi</code>			698, 701, 704, 743, 765, 903, 907
	523, 585, 588, 590	<code>\endinput</code>	81, 428
<code>\AtBegShi@PatchMemoir</code>		<code>\endpicture</code>	371
	592, 681, 684, 686	<code>\errmessage</code>	171
<code>\AtBegShi@Shipout</code>	197, 325		
<code>\AtBegShi@Test</code>	202, 206	F	
<code>\AtBegShi@vorigin</code> .	343, 346, 394, 413	<code>\fill</code>	25
B		G	
<code>\baselineskip</code>	255, 450, 494	<code>\g@addto@macro</code>	296
<code>\begin</code>	11, 50, 905	<code>\gdef</code>	224, 266, 267, 268, 327
<code>\body</code>	711, 715	<code>\GPTorg@shipout</code>	
<code>\box</code>	47, 229, 235, 402, 452, 496,		469, 470, 544, 545, 637, 638
	505, 576, 612, 667, 671, 778,		
	811, 812, 814, 815, 826, 883, 907	H	
C		<code>\hbox</code>	45, 199, 377,
<code>\catcode</code> .	58, 59, 60, 61, 62, 63, 64,		390, 406, 805, 817, 818, 874, 883
	72, 86, 87, 88, 89, 90, 91, 92, 93,	<code>\hrule</code>	821, 882, 884
	94, 95, 96, 97, 98, 119, 120, 123,	<code>\hspace</code>	897, 900
	124, 125, 126, 130, 131, 132,	<code>\hss</code>	421, 878
	133, 137, 139, 191, 192, 194,	<code>\ht</code>	252, 384, 395, 414
	195, 239, 240, 691, 692, 693,		
	694, 729, 738, 751, 752, 753,	I	
	754, 755, 756, 757, 758, 759, 760	<code>\ifAtBegShi@Discarded</code> .	183, 225, 302
<code>\circle</code>	8	<code>\ifcase</code>	162, 349, 621
<code>\color</code>	16, 876	<code>\ifdim</code>	207
<code>\colorbox</code>	898	<code>\ifnum</code>	208, 485, 567, 653, 731, 739
		<code>\ifpdf</code>	341
		<code>\ifvbox</code>	664

<code>\ifvoid</code>	216, 305, 442, 499, 530, 574, 599, 611, 660	<code>\pdfvorigin</code>	343
<code>\ifx</code>	66, 69, 73, 99, 107, 110, 152, 160, 162, 165, 190, 244, 288, 330, 336, 350, 353, 364, 426, 459, 460, 461, 463, 465, 467, 469, 471, 473, 476, 536, 538, 540, 542, 544, 546, 548, 551, 553, 558, 620, 621, 624, 631, 633, 635, 637, 639, 641, 644, 695, 698, 701, 704, 743	<code>\picture</code>	362
<code>\ignorespaces</code>	368, 379	<code>\protect</code>	219, 231, 234
<code>\immediate</code>	75, 101, 770	<code>\ProvidesFile</code>	865
<code>\input</code>	153, 154, 337, 744, 769	<code>\ProvidesPackage</code>	70, 116
<code>\iterate</code>	712, 714, 716	<code>\put</code>	8, 17, 18
K			
<code>\kern</code>	200, 392, 394, 395, 411, 413, 414, 873, 875	R	
L			
<code>\lastkern</code>	207	<code>\RangeCatcodeInvalid</code>	735, 747, 748, 749, 750
<code>\line</code>	17, 18	<code>\repeat</code>	710, 722, 733, 741
<code>\lineskip</code>	256, 448, 492	<code>\RequirePackage</code> ...	156, 157, 339, 866
<code>\lineskiplimit</code>	257, 449, 493	<code>\RestoreCatcodes</code> ..	724, 727, 728, 762
<code>\LoadCommand</code>	744, 761	<code>\rlap</code>	391, 407, 410
<code>\loop</code>	710, 726, 737	<code>\rule</code>	877, 899
<code>\ltx@cclv</code>	576	S	
<code>\ltx@ifUndefined</code>	316	<code>\section</code>	12, 908
<code>\ltx@newif</code>	183	<code>\setbox</code>	45, 199, 204, 229, 246, 254, 377, 390, 406, 439, 482, 501, 527, 564, 576, 596, 650, 662, 780, 805, 826, 871
<code>\ltx@one</code>	163, 166, 627	<code>\shipout</code>	324, 325, 463, 464, 538, 539, 631, 632, 795, 802, 806, 809, 812, 815, 818, 821, 830, 836, 843, 845, 859, 906, 907
<code>\ltx@zero</code>	168, 186, 246, 251, 252, 253, 258, 291, 292, 377, 384, 385, 386, 622, 625	<code>\space</code>	172, 318, 854
M			
<code>\makeatletter</code>	888	<code>\special</code>	46
<code>\makeatother</code>	895	T	
<code>\mem@oldshipout</code>	473, 474, 548, 549, 605, 612, 614, 670	<code>\Test</code>	746, 764
<code>\mem@shipi</code>	595, 620, 649, 652	<code>\testmsg</code>	772, 794, 801, 804, 808, 811, 814, 817, 820, 823, 835, 842, 844, 849, 854
<code>\mem@shipii</code> ...	602, 621, 624, 657, 659	<code>\THBorg@shipout</code>	471, 472, 546, 547, 639, 640
<code>\MessageBreak</code>	283, 307, 318	<code>\the</code> 123, 124, 125, 126, 137, 292, 729, 786	
<code>\msg</code>	770, 771, 773, 774, 786, 790, 833, 837, 840, 846, 856	<code>\TMP@EnsureCode</code> 134, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150	
N			
<code>\NeedsTeXFormat</code>	864	<code>\toks</code>	291, 292
<code>\newbox</code>	777	<code>\trimmarks</code>	606, 615, 663
<code>\newpage</code>	13, 22, 27, 32, 52	<code>\typeout</code>	890
<code>\next</code>	716, 718, 720	U	
<code>\null</code>	794, 795, 823, 830, 844, 845, 854, 859, 906	<code>\unitlength</code>	366
<code>\number</code>	201, 480, 562, 648	<code>\unvbox</code>	607, 616, 665
P			
<code>\p@</code>	200	<code>\usepackage</code> 3, 4, 5, 38, 39, 869, 887, 904	
<code>\PackageInfo</code>	78	V	
<code>\pagecolor</code>	867	<code>\vbox</code> .. 246, 254, 393, 412, 455, 501, 605, 614, 662, 820, 821, 871, 872	
<code>\paperheight</code>	8, 17, 18, 899	<code>\vfill</code>	851
<code>\paperwidth</code>	8, 17, 18, 897, 900	<code>\void</code>	778, 780, 802, 836, 843
<code>\par</code>	24	<code>\VoidBox</code>	824, 829, 831
<code>\pdfhorigin</code>	342	<code>\voidbox</code>	777, 778, 780
		<code>\vrule</code>	805, 818, 883
		<code>\vspace</code>	25
		<code>\vss</code>	399, 418, 880
W			
<code>\wd</code>	251, 406	X	
<code>\write</code>	75, 101, 770	<code>\X</code>	239

$\backslash x$	65, 66, 69,	Z
		74, 78, 80, 100, 105, 115, 121, 129	$\backslash z@$ 448, 449, 450