The **atbegshi** package

Heiko Oberdiek

<*heiko.oberdiek at googlemail.com*>

2011/01/30 v1.15

Abstract

This package is a modern reimplementation of package everyshi without the burden of compatibility. It makes use of ε-TeX's if available. Both LaTeX and plain TeX are supported.

Contents

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 LaTeX'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 pdfTeX 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 LaTeX 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 ⟨/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 catched 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:

$$\texttt{\textbackslash shipout\textbackslash box0} \Rightarrow \texttt{\textbackslash setbox\textbackslash mybox=\textbackslash box0}\ \boxed{\texttt{\textbackslash @test}}$$

However direct boxes can have arbitrary contents with lots of other assignments. It would be quite inpredictable 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:

$$\texttt{\textbackslash shipout\textbackslash hbox\{Hello World\}}$$
$$\Rightarrow \texttt{\textbackslash setbox\textbackslash mybox=\textbackslash hbox\{}\ \boxed{\texttt{\textbackslash @test}}\ \texttt{Hello World\}}$$

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 accidently 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 \lastkern method

And for TeX without ε-TeX using the \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 LaTeX.

```
57 \begingroup\catcode61\catcode48\catcode32=10\relax%
58   \catcode13=5 % ^^M
59   \endlinechar=13 %
60   \catcode35=6 % #
61   \catcode39=12 % '
62   \catcode44=12 % ,
63   \catcode45=12 % -
64   \catcode46=12 % .
65   \catcode58=12 % :
66   \catcode64=11 % @
67   \catcode123=1 % {
68   \catcode125=2 % }
69 \expandafter\let\expandafter\x\csname ver@atbegshi.sty\endcsname
70 \ifx\x\relax % plain-TeX, first loading
71 \else
72   \def\empty{}%
73   \ifx\x\empty % LaTeX, first loading,
74     % variable is initialized, but \ProvidesPackage not yet seen
75   \else
76     \expandafter\ifx\csname PackageInfo\endcsname\relax
77       \def\x#1#2{%
78         \immediate\write-1{Package #1 Info: #2.}%
79       }%
80     \else
81       \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
82     \fi
83     \x{atbegshi}{The package is already loaded}%
84     \aftergroup\endinput
85   \fi
```

```
 86   \fi
 87 \endgroup%
```

Package identification:

```
 88 \begingroup\catcode61\catcode48\catcode32=10\relax%
 89   \catcode13=5 % ^^M
 90   \endlinechar=13 %
 91   \catcode35=6 % #
 92   \catcode39=12 % '
 93   \catcode40=12 % (
 94   \catcode41=12 % )
 95   \catcode44=12 % ,
 96   \catcode45=12 % -
 97   \catcode46=12 % .
 98   \catcode47=12 % /
 99   \catcode58=12 % :
100   \catcode64=11 % @
101   \catcode91=12 % [
102   \catcode93=12 % ]
103   \catcode123=1 % {
104   \catcode125=2 % }
105   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
106     \def\x#1#2#3[#4]{\endgroup
107       \immediate\write-1{Package: #3 #4}%
108       \xdef#1{#4}%
109     }%
110   \else
111     \def\x#1#2[#3]{\endgroup
112       #2[{#3}]%
113       \ifx#1\@undefined
114         \xdef#1{#3}%
115       \fi
116       \ifx#1\relax
117         \xdef#1{#3}%
118       \fi
119     }%
120   \fi
121 \expandafter\x\csname ver@atbegshi.sty\endcsname
122 \ProvidesPackage{atbegshi}%
123   [2011/01/30 v1.15 At begin shipout hook (HO)]%
```

3.2   Catcodes

```
124 \begingroup\catcode61\catcode48\catcode32=10\relax%
125   \catcode13=5 % ^^M
126   \endlinechar=13 %
127   \catcode123=1 % {
128   \catcode125=2 % }
129   \catcode64=11 % @
130   \def\x{\endgroup
131     \expandafter\edef\csname AtBegShi@AtEnd\endcsname{%
132       \endlinechar=\the\endlinechar\relax
133       \catcode13=\the\catcode13\relax
134       \catcode32=\the\catcode32\relax
135       \catcode35=\the\catcode35\relax
136       \catcode61=\the\catcode61\relax
137       \catcode64=\the\catcode64\relax
138       \catcode123=\the\catcode123\relax
139       \catcode125=\the\catcode125\relax
140     }%
141   }%
142 \x\catcode61\catcode48\catcode32=10\relax%
143 \catcode13=5 % ^^M
```

11

```
144 \endlinechar=13 %
145 \catcode35=6 % #
146 \catcode64=11 % @
147 \catcode123=1 % {
148 \catcode125=2 % }
149 \def\TMP@EnsureCode#1#2{%
150   \edef\AtBegShi@AtEnd{%
151     \AtBegShi@AtEnd
152     \catcode#1=\the\catcode#1\relax
153   }%
154   \catcode#1=#2\relax
155 }
156 \TMP@EnsureCode{40}{12}% (
157 \TMP@EnsureCode{41}{12}% )
158 \TMP@EnsureCode{44}{12}% ,
159 \TMP@EnsureCode{45}{12}% -
160 \TMP@EnsureCode{47}{12}% /
161 \TMP@EnsureCode{46}{12}% .
162 \TMP@EnsureCode{58}{12}% :
163 \TMP@EnsureCode{91}{12}% [
164 \TMP@EnsureCode{93}{12}% ]
165 \TMP@EnsureCode{94}{7}% ^(superscript)
166 \TMP@EnsureCode{96}{12}% `
167 \edef\AtBegShi@AtEnd{\AtBegShi@AtEnd\noexpand\endinput}
```

3.3   Preparations

```
168 \begingroup\expandafter\expandafter\expandafter\endgroup
169 \expandafter\ifx\csname RequirePackage\endcsname\relax
170   \def\TMP@RequirePackage#1[#2]{%
171     \begingroup\expandafter\expandafter\expandafter\endgroup
172     \expandafter\ifx\csname ver@#1.sty\endcsname\relax
173       \input #1.sty\relax
174     \fi
175   }%
176   \TMP@RequirePackage{infwarerr}[2007/09/09]%
177   \TMP@RequirePackage{ltxcmds}[2010/03/01]%
178 \else
179   \RequirePackage{infwarerr}[2007/09/09]%
180   \RequirePackage{ltxcmds}[2010/03/01]%
181 \fi
```

\AtBegShi@CheckDefinable

```
182 \begingroup\expandafter\expandafter\expandafter\endgroup
183 \expandafter\ifx\csname @ifdefinable\endcsname\relax
184   \def\AtBegShi@CheckDefinable#1{%
185     \ifcase\ifx#1\relax
186             \ltx@one
187           \else
188             \ifx#1\@undefined
189               \ltx@one
190             \else
191               \ltx@zero
192             \fi
193           \fi
194     \@PackageError{atbegshi}{%
195       \string#1\space is already defined%
196     }\@ehd
197   \fi
198 }%
199 \else
200   \def\AtBegShi@CheckDefinable#1{%
201     \@ifdefinable{#1}{}%
202 }%
```

```
203 \fi

204 \ltx@newif\ifAtBegShi@Discarded
```

\AtBeginShipoutDiscard

```
205 \AtBegShi@CheckDefinable\AtBeginShipoutDiscard
206 \def\AtBeginShipoutDiscard{%
207   \deadcycles=\ltx@zero
208   \global\AtBegShi@Discardedtrue
209 }

210 \begingroup\expandafter\expandafter\expandafter\endgroup
211 \expandafter\ifx\csname currentgrouplevel\endcsname\relax
212   \catcode'X=9  % ignore
213   \catcode'E=14 % comment
214 \else
215   \catcode'X=14 % comment
216   \catcode'E=9  % ignore
217 \fi
```

\AtBegShi@Shipout

```
218 \def\AtBegShi@Shipout{%
219 X \begingroup
220 X \setbox\AtBeginShipoutBox=\hbox\bgroup
221 X \kern\p@
222 E \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
223   \afterassignment\AtBegShi@Test
224 X \global
225   \setbox\AtBeginShipoutBox=%
226 }
```

\AtBegShi@Test

```
227 \def\AtBegShi@Test{%
228 X \ifdim\lastkern=0pt %
229 E \ifnum\AtBegShi@GroupLevel<\currentgrouplevel
230     \expandafter\aftergroup
231   \fi
232   \AtBegShi@Output
233 }
```

\AtBegShi@Output

```
234 \def\AtBegShi@Output{%
235 X \egroup
236 X \endgroup
237   \ifvoid\AtBeginShipoutBox
238     \@PackageWarning{atbegshi}{Ignoring void shipout box}%
239   \else
240     \let\AtBegShi@OrgProtect\protect
241     \csname set@typeset@protect\endcsname
242     \global\AtBegShi@Discardedfalse
243     \AtBegShi@Hook
244     \expandafter\gdef\expandafter\AtBegShi@HookNext
245     \expandafter{\expandafter}%
246     \AtBegShi@HookNext
247     \ifAtBegShi@Discarded
248       \@PackageInfoNoLine{atbegshi}{Shipout page discarded}%
249       \global\AtBegShi@Discardedfalse
250       \begingroup
251         \setbox\AtBeginShipoutBox\box\AtBeginShipoutBox
252       \endgroup
253       \let\protect\AtBegShi@OrgProtect
254     \else
```

```
255        \AtBegShi@First
256        \let\protect\AtBegShi@OrgProtect
257        \AtBeginShipoutOriginalShipout\box\AtBeginShipoutBox
258      \fi
259    \fi
260 }

261 \catcode`\X=11 %
262 \catcode`\E=11 %
```

\AtBegShi@First

```
263 \def\AtBegShi@First{%
264   \begingroup
265     \def\@empty{}%
266     \ifx\AtBegShi@HookFirst\@empty
267     \else
268       \setbox\ltx@zero=\vbox{%
269         \begingroup
270           \AtBegShi@HookFirst
271         \endgroup
272       }%
273       \wd\ltx@zero=0pt %
274       \ht\ltx@zero=0pt %
275       \dp\ltx@zero=0pt %
276       \global\setbox\AtBeginShipoutBox=\vbox{%
277         \baselineskip 0pt\relax
278         \lineskip 0pt\relax
279         \lineskiplimit 0pt\relax
280         \copy\ltx@zero
281         \copy\AtBeginShipoutBox
282       }%
283     \fi
284     \global\let\AtBegShi@First\@empty
285     \global\let\AtBeginShipoutFirst\AtBegShi@FirstDisabled
286   \endgroup
287 }
```

\AtBegShi@Hook

```
288 \gdef\AtBegShi@Hook{}
```

\AtBegShi@HookNext

```
289 \gdef\AtBegShi@HookNext{}
```

\AtBegShi@HookFirst

```
290 \gdef\AtBegShi@HookFirst{}
```

\AtBeginShipout

```
291 \AtBegShi@CheckDefinable\AtBeginShipout
292 \def\AtBeginShipout{%
293   \AtBegShi@AddHook\AtBegShi@Hook
294 }
```

\AtBeginShipoutNext

```
295 \AtBegShi@CheckDefinable\AtBeginShipoutNext
296 \def\AtBeginShipoutNext{%
297   \AtBegShi@AddHook\AtBegShi@HookNext
298 }
```

\AtBeginShipoutFirst

```
299 \AtBegShi@CheckDefinable\AtBeginShipoutFirst
300 \def\AtBeginShipoutFirst{%
301   \AtBegShi@AddTo\AtBegShi@HookFirst
302 }
```

14

`\AtBegShi@FirstDisabled`

```
303 \long\def\AtBegShi@FirstDisabled#1{%
304   \@PackageWarning{atbegshi}{%
305     First page is already shipped out, ignoring\MessageBreak
306     \string\AtBeginShipoutFirst
307   }%
308 }
```

`\AtBegShi@AddTo`

```
309 \begingroup\expandafter\expandafter\expandafter\endgroup
310 \expandafter\ifx\csname g@addto@macro\endcsname\relax
311   \long\def\AtBegShi@AddTo#1#2{%
312     \begingroup
313       \toks\ltx@zero\expandafter{#1#2}%
314       \xdef#1{\the\toks\ltx@zero}%
315     \endgroup
316   }%
317 \else
318   \let\AtBegShi@AddTo\g@addto@macro
319 \fi
```

`\AtBegShi@AddHook`

```
320 \long\def\AtBegShi@AddHook#1#2{%
321   \AtBegShi@AddTo#1{\AtBegShi@Item{#2}}%
322 }
```

`\AtBegShi@Item`

```
323 \long\def\AtBegShi@Item#1{%
324   \ifAtBegShi@Discarded
325   \else
326     #1%
327     \ifAtBegShi@Discarded
328     \else
329       \ifvoid\AtBeginShipoutBox
330         \@PackageWarning{atbegshi}{%
331           Shipout box was voided by hook,\MessageBreak
332           ignoring shipout box%
333         }%
334         \AtBeginShipoutDiscard
335       \fi
336     \fi
337   \fi
338 }
```

`\AtBeginShipoutInit`

```
339 \AtBegShi@CheckDefinable\AtBeginShipoutInit
340 \def\AtBeginShipoutInit{%
341   \ltx@IfUndefined{newbox}{%
342     \@PackageError{atbegshi}{%
343       \string\AtBeginShipoutInit\space failed\MessageBreak
344        because of missing \expandafter\string\csname newbox\endcsname
345     }\@ehc
346   }{%
347     \csname newbox\endcsname\AtBeginShipoutBox
348     \AtBegShi@CheckDefinable\AtBeginShipoutOriginalShipout
349     \global\let\AtBeginShipoutOriginalShipout\shipout
350     \global\let\shipout\AtBegShi@Shipout
351   }%
352   \gdef\AtBeginShipoutInit{}%
353 }
```

```
354 \begingroup\expandafter\expandafter\expandafter\endgroup
355 \expandafter\ifx\csname AtBeginDocument\endcsname\relax
356   \AtBeginShipoutInit
357 \else
358   \AtBeginDocument{\AtBeginShipoutInit}%
359 \fi
```

3.4   Positioning

```
360 \begingroup\expandafter\expandafter\expandafter\endgroup
361 \expandafter\ifx\csname RequirePackage\endcsname\relax
362   \input ifpdf.sty\relax
363 \else
364   \RequirePackage{ifpdf}\relax
365 \fi
366 \ifpdf
367   \def\AtBegShi@horigin{\pdfhorigin}%
368   \def\AtBegShi@vorigin{\pdfvorigin}%
369 \else
370   \def\AtBegShi@horigin{72.27pt}%
371   \def\AtBegShi@vorigin{72.27pt}%
372 \fi
373 \begingroup
374 \ifcase
375     \expandafter\ifx\csname picture\endcsname\relax
376       1%
377     \else
378       \expandafter\ifx\csname endpicture\endcsname\relax
379         1%
380       \else
381         0%
382       \fi
383     \fi
384   \endgroup
385   \def\AtBegShi@BeginPicture{%
386     \begingroup
387     \picture(0,0)\relax
388     \begingroup\expandafter\expandafter\expandafter\endgroup
389     \expandafter\ifx\csname unitlength\endcsname\relax
390     \else
391       \unitlength=1pt\relax
392     \fi
393     \ignorespaces
394   }%
395   \def\AtBegShi@EndPicture{%
396     \endpicture
397     \endgroup
398   }%
399 \else
400   \endgroup
401   \def\AtBegShi@BeginPicture{%
402     \setbox\ltx@zero=\hbox\bgroup
403     \begingroup
404     \ignorespaces
405   }%
406   \def\AtBegShi@EndPicture{%
407     \endgroup
408     \egroup
409     \ht\ltx@zero=0pt\relax
410     \dp\ltx@zero=0pt\relax
411     \copy\ltx@zero
412   }%
```

```
413 \fi
414 \def\AtBeginShipoutUpperLeft#1{%
415   \global\setbox\AtBeginShipoutBox=\hbox{%
416     \rlap{%
417       \kern-\AtBegShi@horigin\relax
418       \vbox to 0pt{%
419         \kern-\AtBegShi@vorigin\relax
420         \kern-\ht\AtBeginShipoutBox
421         \AtBegShi@BeginPicture
422         #1%
423         \AtBegShi@EndPicture
424         \vss
425       }%
426     }%
427     \box\AtBeginShipoutBox
428   }%
429 }
430 \def\AtBeginShipoutUpperLeftForeground#1{%
431   \global\setbox\AtBeginShipoutBox=\hbox to \wd\AtBeginShipoutBox{%
432     \rlap{%
433       \copy\AtBeginShipoutBox
434     }%
435     \rlap{%
436       \kern-\AtBegShi@horigin\relax
437       \vbox to 0pt{%
438         \kern-\AtBegShi@vorigin\relax
439         \kern-\ht\AtBeginShipoutBox
440         \AtBegShi@BeginPicture
441         #1%
442         \AtBegShi@EndPicture
443         \vss
444       }%
445     }%
446     \hss
447   }%
448 }
```

3.5 Patches

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

```
449 \def\AtBegShi@AbortIfUndefined#1{%
450   \begingroup\expandafter\expandafter\expandafter\endgroup
451   \expandafter\ifx\csname#1\endcsname\relax
452     \expandafter\AtBegShi@AtEnd
453   \fi
454 }
455 \AtBegShi@AbortIfUndefined{currentgrouplevel}%
456 \AtBegShi@AbortIfUndefined{AtBeginDocument}%
457 \AtBegShi@AbortIfUndefined{@ifpackageloaded}%
458 \AtBegShi@AbortIfUndefined{@ifclassloaded}%
```

3.5.1 Package **crop**

Fix of method and box.

```
459 \def\AtBegShi@PatchCrop{%
460   \begingroup
461     \def\AtBegShi@Crop@shipout{%
462       \afterassignment\CROP@ship
463       \setbox\@cclv=%
464     }%
465     \def\AtBegShi@Crop@ship{%
```

```
466     \ifvoid\@cclv
467       \expandafter\aftergroup
468     \fi
469     \CROP@@ship
470   }%
471   \def\AtBegShi@Crop@shiplist{%
472     \lineskip\z@
473     \lineskiplimit\z@
474     \baselineskip\z@
475     \CROP@kernel
476     \box\@cclv
477   }%
478   \def\AtBegShi@Crop@@ship{%
479     \CROP@shipout\vbox{%
480       \CROP@shiplist
481     }%
482   }%
483   \ifx\AtBegShi@Crop@ship\CROP@ship
484     \ifx\AtBegShi@Crop@shiplist\CROP@shiplist
485       \ifx\AtBegShi@Crop@@ship\CROP@@ship
486         \let\AtBegShi@found\relax
487         \ifx\shipout\AtBegShi@Crop@shipout
488           \def\AtBegShi@found{\shipout}%
489         \else\ifx\AtBeginShipoutOriginalShipout\AtBegShi@Crop@shipout
490           \def\AtBegShi@found{\AtBeginShipoutOriginalShipout}%
491         \else\ifx\@EveryShipout@Org@Shipout\AtBegShi@Crop@shipout
492           \def\AtBegShi@found{\@EveryShipout@Org@Shipout}%
493         \else\ifx\GPTorg@shipout\AtBegShi@Crop@shipout
494           \def\AtBegShi@found{\GPTorg@shipout}%
495         \else\ifx\THBorg@shipout\AtBegShi@Crop@shipout
496           \def\AtBegShi@found{\THBorg@shipout}%
497         \else\ifx\mem@oldshipout\AtBegShi@Crop@shipout
498           \def\AtBegShi@found{\mem@oldshipout}%
499         \fi\fi\fi\fi\fi\fi
500         \ifx\AtBegShi@found\relax
501         \else
502           \expandafter\endgroup
503           \expandafter\def\AtBegShi@found{%
504             \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
505             \afterassignment\CROP@ship
506             \setbox\AtBeginShipoutBox=%
507           }%
508           \def\CROP@ship{%
509             \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
510             \else
511               \expandafter\aftergroup
512             \fi
513             \CROP@@ship
514           }%
515           \def\CROP@shiplist{%
516             \lineskip 0pt\relax
517             \lineskiplimit 0pt\relax
518             \baselineskip 0pt\relax
519             \CROP@kernel
520             \box\AtBeginShipoutBox
521           }%
522           \def\CROP@@ship{%
523             \ifvoid\AtBeginShipoutBox
524             \else
525               \setbox\AtBeginShipoutBox=\vbox{%
526                 \CROP@shiplist
527               }%
```

```
528              \expandafter\CROP@shipout
529                \expandafter\box
530                \expandafter\AtBeginShipoutBox
531              \fi
532            }%
533            \@PackageInfoNoLine{atbegshi}{Package 'crop' patched}%
534            \begingroup
535          \fi
536        \fi
537      \fi
538    \fi
539  \endgroup
540  \let\AtBegShi@PatchCrop\relax
541 }
542 \@ifpackageloaded{crop}{%
543   \AtBegShi@PatchCrop
544 }{%
545   \AtBeginDocument{\AtBegShi@PatchCrop}%
546 }
```

3.5.2 Package everyshi

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

```
547 \def\AtBegShi@PatchEveryshi{%
548   \begingroup
549    \long\def\AtBegShi@Everyshi@shipout{%
550      \afterassignment\@EveryShipout@Test
551      \global\setbox\@cclv= %
552    }%
553    \long\def\AtBegShi@Everyshi@Test{%
554      \ifvoid\@cclv\relax
555        \aftergroup\@EveryShipout@Output
556      \else
557        \@EveryShipout@Output
558      \fi
559    }%
560    \ifx\AtBegShi@Everyshi@Test\@EveryShipout@Test
561      \let\AtBegShi@found\relax
562      \ifx\shipout\AtBegShi@Everyshi@shipout
563        \def\AtBegShi@found{\shipout}%
564      \else\ifx\AtBeginShipoutOriginalShipout\AtBegShi@Everyshi@shipout
565        \def\AtBegShi@found{\AtBeginShipoutOriginalShipout}%
566      \else\ifx\CROP@shipout\AtBegShi@Everyshi@shipout
567        \def\AtBegShi@found{\CROP@shipout}%
568      \else\ifx\GPTorg@shipout\AtBegShi@Everyshi@shipout
569        \def\AtBegShi@found{\GPTorg@shipout}%
570      \else\ifx\THBorg@shipout\AtBegShi@Everyshi@shipout
571        \def\AtBegShi@found{\THBorg@shipout}%
572      \else\ifx\mem@oldshipout\AtBegShi@Everyshi@shipout
573        \def\AtBegShi@found{\mem@oldshipout}%
574      \else
575        \expandafter\ifx\csname @EveryShipout@Org@Shipout\endcsname
576                    \relax
577          \ifx\@EveryShipout@Shipout\AtBegShi@Everyshi@shipout
578            \def\AtBegShi@found{\@EveryShipout@Shipout}%
579          \fi
580        \fi
581      \fi\fi\fi\fi\fi\fi
582      \ifx\AtBegShi@found\relax
583      \else
584        \expandafter\endgroup
585        \expandafter\def\AtBegShi@found{%
```

```
586         \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
587         \afterassignment\@EveryShipout@Test
588         \setbox\AtBeginShipoutBox=%
589       }%
590       \def\@EveryShipout@Test{%
591         \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
592         \else
593           \expandafter\aftergroup
594         \fi
595         \AtBegShi@Everyshi@Output
596       }%
597       \def\AtBegShi@Everyshi@Output{%
598         \ifvoid\AtBeginShipoutBox
599         \else
600           \global\setbox\ltx@cclv\box\AtBeginShipoutBox
601           \expandafter\@EveryShipout@Output
602         \fi
603       }%
604       \@PackageInfoNoLine{atbegshi}{Package 'everyshi' patched}%
605       \begingroup
606     \fi
607   \fi
608   \endgroup
609   \let\AtBegShi@PatchEveryshi\relax
610 }
611 \@ifpackageloaded{everyshi}{%
612   \AtBegShi@PatchEveryshi
613 }{%
614   \AtBeginDocument{\AtBegShi@PatchEveryshi}%
615 }
```

3.5.3   Class **memoir**

Fix of method and box.

```
616 \def\AtBegShi@PatchMemoir{%
617   \begingroup
618     \def\AtBegShi@Memoir@shipout{%
619       \afterassignment\mem@shipi
620       \setbox\@cclv=%
621     }%
622     \def\AtBegShi@Memoir@shipi{%
623       \ifvoid\@cclv
624         \expandafter\aftergroup
625       \fi
626       \mem@shipii
627     }%
628     \def\AtBegShi@Memoir@shipiiA{%
629       \mem@oldshipout\vbox{%
630         \trimmarks
631         \unvbox\@cclv
632       }%
633     }%
634     \def\AtBegShi@Memoir@shipiiB{%
635       \ifvoid\@cclv
636         \mem@oldshipout\box\@cclv
637       \else
638         \mem@oldshipout\vbox{%
639           \trimmarks
640           \unvbox\@cclv
641         }%
642       \fi
643     }%
```

```
644    \ifx\AtBegShi@Memoir@shipi\mem@shipi
645      \ifcase\ifx\AtBegShi@Memoir@shipiiA\mem@shipii
646            \ltx@zero
647          \else
648            \ifx\AtBegShi@Memoir@shipiiB\mem@shipii
649              \ltx@zero
650            \else
651              \ltx@one
652            \fi
653          \fi
654      \let\AtBegShi@found\relax
655      \ifx\shipout\AtBegShi@Memoir@shipout
656        \def\AtBegShi@found{\shipout}%
657      \else\ifx\AtBeginShipoutOriginalShipout\AtBegShi@Memoir@shipout
658        \def\AtBegShi@found{\AtBeginShipoutOriginalShipout}%
659      \else\ifx\CROP@shipout\AtBegShi@Memoir@shipout
660        \def\AtBegShi@found{\CROP@shipout}%
661      \else\ifx\GPTorg@shipout\AtBegShi@Memoir@shipout
662        \def\AtBegShi@found{\GPTorg@shipout}%
663      \else\ifx\THBorg@shipout\AtBegShi@Memoir@shipout
664        \def\AtBegShi@found{\THBorg@shipout}%
665      \else\ifx\@EveryShipout@Org@Shipout\AtBegShi@Memoir@shipout
666        \def\AtBegShi@found{\@EveryShipout@Org@Shipout}%
667      \fi\fi\fi\fi\fi\fi
668      \ifx\AtBegShi@found\relax
669      \else
670        \expandafter\endgroup
671        \expandafter\def\AtBegShi@found{%
672          \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
673          \afterassignment\mem@shipi
674          \setbox\AtBeginShipoutBox=%
675        }%
676        \def\mem@shipi{%
677          \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
678          \else
679            \expandafter\aftergroup
680          \fi
681          \mem@shipii
682        }%
683        \def\mem@shipii{%
684          \ifvoid\AtBeginShipoutBox
685          \else
686            \setbox\AtBeginShipoutBox=\vbox{%
687              \trimmarks
688              \ifvbox\AtBeginShipoutBox
689                \unvbox\AtBeginShipoutBox
690              \else
691                \box\AtBeginShipoutBox
692              \fi
693            }%
694            \expandafter\mem@oldshipout
695            \expandafter\box
696            \expandafter\AtBeginShipoutBox
697          \fi
698        }%
699        \@PackageInfoNoLine{atbegshi}{Class `memoir' patched}%
700        \begingroup
701      \fi
702    \fi
703    \fi
704  \endgroup
705  \let\AtBegShi@PatchMemoir\relax
```

```
706 }
707 \@ifclassloaded{memoir}{%
708   \AtBegShi@PatchMemoir
709 }{%
710   \AtBeginDocument{\AtBegShi@PatchMemoir}%
711 }

712 \AtBegShi@AtEnd%
713 ⟨/package⟩
```

4 Test

4.1 Catcode checks for loading

```
714 ⟨*test1⟩
```

```
715 \catcode`\{=1 %
716 \catcode`\}=2 %
717 \catcode`\#=6 %
718 \catcode`\@=11 %
719 \expandafter\ifx\csname count@\endcsname\relax
720   \countdef\count@=255 %
721 \fi
722 \expandafter\ifx\csname @gobble\endcsname\relax
723   \long\def\@gobble#1{}%
724 \fi
725 \expandafter\ifx\csname @firstofone\endcsname\relax
726   \long\def\@firstofone#1{#1}%
727 \fi
728 \expandafter\ifx\csname loop\endcsname\relax
729   \expandafter\@firstofone
730 \else
731   \expandafter\@gobble
732 \fi
733 {%
734   \def\loop#1\repeat{%
735     \def\body{#1}%
736     \iterate
737   }%
738   \def\iterate{%
739     \body
740       \let\next\iterate
741     \else
742       \let\next\relax
743     \fi
744     \next
745   }%
746   \let\repeat=\fi
747 }%
748 \def\RestoreCatcodes{}
749 \count@=0 %
750 \loop
751   \edef\RestoreCatcodes{%
752     \RestoreCatcodes
753     \catcode\the\count@=\the\catcode\count@\relax
754   }%
755 \ifnum\count@<255 %
756   \advance\count@ 1 %
757 \repeat
758
759 \def\RangeCatcodeInvalid#1#2{%
760   \count@=#1\relax
761   \loop
```

22

```
762    \catcode\count@=15 %
763  \ifnum\count@<#2\relax
764    \advance\count@ 1 %
765  \repeat
766 }
767 \def\RangeCatcodeCheck#1#2#3{%
768  \count@=#1\relax
769  \loop
770    \ifnum#3=\catcode\count@
771    \else
772      \errmessage{%
773        Character \the\count@\space
774        with wrong catcode \the\catcode\count@\space
775        instead of \number#3%
776      }%
777    \fi
778  \ifnum\count@<#2\relax
779    \advance\count@ 1 %
780  \repeat
781 }
782 \def\space{ }
783 \expandafter\ifx\csname LoadCommand\endcsname\relax
784  \def\LoadCommand{\input atbegshi.sty\relax}%
785 \fi
786 \def\Test{%
787  \RangeCatcodeInvalid{0}{47}%
788  \RangeCatcodeInvalid{58}{64}%
789  \RangeCatcodeInvalid{91}{96}%
790  \RangeCatcodeInvalid{123}{255}%
791  \catcode`\@=12 %
792  \catcode`\\=0 %
793  \catcode`\%=14 %
794  \LoadCommand
795  \RangeCatcodeCheck{0}{36}{15}%
796  \RangeCatcodeCheck{37}{37}{14}%
797  \RangeCatcodeCheck{38}{47}{15}%
798  \RangeCatcodeCheck{48}{57}{12}%
799  \RangeCatcodeCheck{58}{63}{15}%
800  \RangeCatcodeCheck{64}{64}{12}%
801  \RangeCatcodeCheck{65}{90}{11}%
802  \RangeCatcodeCheck{91}{91}{15}%
803  \RangeCatcodeCheck{92}{92}{0}%
804  \RangeCatcodeCheck{93}{96}{15}%
805  \RangeCatcodeCheck{97}{122}{11}%
806  \RangeCatcodeCheck{123}{255}{15}%
807  \RestoreCatcodes
808 }
809 \Test
810 \csname @@end\endcsname
811 \end
812 ⟨/test1⟩
813 ⟨*test2⟩
814 \input atbegshi.sty\relax
815 \def\msg#{\immediate\write16}
816 \msg{File: atbegshi-test2.tex 2011/01/30 v1.15 Test file for plain-TeX}
817 \def\testmsg#1#2{%
818  \msg{}%
819  \msg{*** Test with box (#1), expected page output [#2]}% hash-ok
820 }
821
822 \newbox\voidbox
823 \def\void{\box\voidbox}
```

```
824 \begingroup
825   \setbox\voidbox=\void
826 \endgroup
827
828 \count0=0\relax
829 \AtBeginShipout{%
830   \global\advance\count0 by 1\relax
831   \msg{* Inside \string\AtBeginShipout: [\the\count0]}%
832 }
833
834 \AtBeginShipoutFirst{%
835   \msg{* Inside \string\AtBeginShipoutFirst}%
836   Hello World%
837 }
838
839 \testmsg{\string\null}{1}
840 \shipout\null
841
842 \AtBeginShipoutFirst{%
843   This is too late%
844 }
845
846 \testmsg{void}{}
847 \shipout\void
848
849 \testmsg{\string\copy255 (not void)}{2}
850 \setbox255\hbox{\vrule height 10bp width 10bp}
851 \shipout\copy255 %
852
853 \testmsg{\string\copy255 (again)}{3}
854 \shipout\copy255 %
855
856 \testmsg{\string\box255}{4}
857 \shipout\box255 %
858
859 \testmsg{\string\box255 (again)}{}
860 \shipout\box255 %
861
862 \testmsg{\string\hbox}{5}
863 \shipout\hbox{\vrule height 5bp width 20bp}
864
865 \testmsg{\string\vbox}{6}
866 \shipout\vbox{\hrule height 20bp width 5bp}
867
868 \testmsg{\string\null, voided by hook}{}
869 \def\VoidBox{%
870   \begingroup
871     \setbox\AtBeginShipoutBox=\box\AtBeginShipoutBox
872   \endgroup
873 }
874 \AtBeginShipout{\VoidBox}
875 \shipout\null
876 \def\VoidBox{}
877
878 \msg{*** \string\begingroup}
879 \begingroup
880   \testmsg{void}{}%
881   \shipout\void
882 \msg{*** \string\endgroup}
883 \endgroup
884
885 \msg{*** \string\begingroup}
```

```
886 \begingroup
887   \testmsg{void}{}%
888   \shipout\void
889   \testmsg{\string\null}{8}%
890   \shipout\null
891 \msg{*** \string\endgroup}
892 \endgroup
893
894 \testmsg{output routine}{9}
895 Hello World
896 \vfill
897 \eject
898
899 \testmsg{\string\null\space(discarded)}{}
900 \AtBeginShipout{%
901   \msg{* Inside \string\AtBeginShipout: DISCARD}%
902   \AtBeginShipoutDiscard
903 }
904 \shipout\null
905
906 \end
907 ⟨/test2⟩

908 ⟨*test3⟩
909 \NeedsTeXFormat{LaTeX2e}
910 \ProvidesFile{atbegshi-test3.tex}[2011/01/30 v1.15 Test file for LaTeX]
911 \RequirePackage{color}
912 \pagecolor{yellow}
913 \documentclass[a5paper,showtrims]{memoir}
914 \usepackage{atbegshi}
915 \AtBeginShipout{%
916   \setbox\AtBeginShipoutBox=\vbox{%
917     \vbox to 0pt{%
918       \kern-1.5in %
919       \hbox to 0pt{%
920         \kern-1.5in %
921         \color{blue}%
922         \rule{1in}{1in}%
923         \hss
924       }%
925       \vss
926     }%
927     \hrule
928     \hbox{\vrule\box\AtBeginShipoutBox\vrule}%
929     \hrule
930   }%
931 }
932 \usepackage{eso-pic}
933 \makeatletter
934 \@ifundefined{@EveryShipout@Init}{%
935   \typeout{Test skipped}%
936   \@@end
937 }{}
938 \@EveryShipout@Init
939 \let\@EveryShipout@Init\relax
940 \makeatother
941 \AddToShipoutPicture{%
942   \hspace{.52\paperwidth}%
943   \colorbox{cyan}{%
944     \rule{0mm}{\paperheight}%
945     \hspace{.48\paperwidth}%
946   }%
947 }
```

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

```
948 \expandafter\let\csname ver@crop.sty\endcsname\relax
949 \usepackage[color=red,cross,a4,center]{crop}
950 \begin{document}
951 \shipout\null
952 \shipout\box\csname voidb@x\endcsname
953 \section{Hello World}
954 \end{document}
955 ⟨/test3⟩
```

5 Installation

5.1 Download

Package. This package is available on CTAN[1]:

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
```

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):

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

---

[1] ftp://ftp.ctan.org/tex-archive/

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 (teTeX, mikTeX, . . .) relies on file name databases, you must refresh these. For example, teTeX users run `texhash` or `mktexlsr`.

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 LaTeX. The `.dtx` chooses its action depending on the format:

plain TeX: Run docstrip and extract the files.

LaTeX: Generate the documentation.

If you insist on using LaTeX for docstrip (really, docstrip does not need LaTeX), 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 pdfLaTeX:

```
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-TeX 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`.

[2010/03/25 v1.12]

- `\AtBeginShipoutNext` can now be used inside `\AtBeginShipoutNext`.

[2010/08/18 v1.13]

- Fixes for `\AtBegShi@CheckDefinable`.

[2010/12/02 v1.14]

- Remove the warning because of void box if the hook calls .

[2011/01/30 v1.15]

- Already loaded package files are not input in plain TeX.

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; plain numbers refer to the code lines where the entry is used.