

# The atbegshi package

Heiko Oberdiek  
<heiko.oberdiek at gmail.com>

2010/03/25 v1.12

## Abstract

This package is a modern reimplementaion of package `everyshi` without the burden of compatibility. It makes use of  $\epsilon$ -TeX's if available. Both L<sup>A</sup>T<sub>E</sub>X and plain T<sub>E</sub>X are supported.

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Examples	4
1.1.1	Example: circle in background	4
1.1.2	Example: adding TrimBox for dvipdfmx	4
<b>2</b>	<b>Method of <code>\shipout</code> overloading</b>	<b>5</b>
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 $\epsilon$ -TeX	6
2.3.2	Without $\epsilon$ -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 $\epsilon$ -TeX	8
2.6.2	Without $\epsilon$ -TeX, traditional way	9
2.6.3	<code>\lastkern</code> method	9
<b>3</b>	<b>Implementation</b>	<b>10</b>
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
<b>4</b>	<b>Test</b>	<b>21</b>
4.1	Catcode checks for loading	21
<b>5</b>	<b>Installation</b>	<b>25</b>
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

<b>6 History</b>	<b>27</b>
[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
[2010/03/25 v1.12]	28
<b>7 Index</b>	<b>28</b>

## 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  $\text{\LaTeX}$  calls `\shipout` in `\@outputpage` (part of its output routine), the meaning of `\protect` is `\noexpand`.  $\text{\LaTeX}$  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<sup>A</sup>T<sub>E</sub>X'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 pdfT<sub>E</sub>X 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<sup>A</sup>T<sub>E</sub>X 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  $\text{\TeX}$  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{}` ( $= \text{\null}$ ) 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  $\text{\TeX}$  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  $\text{\TeX}$ , 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 \@test
```

However direct boxes can have arbitrary contents with lots of other assignments. It would be quite unpredictable if  $\text{\TeX}$  would put `\@test` after the first of such an assignment or after the box specification if the box lacks of assignments. Therefore  $\text{\TeX}$  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 $\varepsilon\text{-TeX}$

With the  $\varepsilon\text{-TeX}$ '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 $\varepsilon\text{-TeX}$

Life becomes complicate without  $\varepsilon\text{-TeX}$ . 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.  $\text{\TeX}$  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  $\varepsilon$ -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  $\varepsilon$ -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 $\varepsilon$ -TeX

Putting the pieces together we get for  $\varepsilon$ -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 $\varepsilon$ -T<sub>E</sub>X, traditional way

And for T<sub>E</sub>X without  $\varepsilon$ -T<sub>E</sub>X:

```

\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 T<sub>E</sub>X without  $\varepsilon$ -T<sub>E</sub>X 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  $\epsilon$ -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<sup>A</sup>T<sub>E</sub>X.

```

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 [2010/03/25 v1.12 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 \expandafter\gdef\expandafter\AtBegShi@HookNext
224 \expandafter{\expandafter}%
225 \AtBegShi@HookNext
226 \ifAtBegShi@Discarded
227 \@PackageInfoNoLine{atbegshi}{Shipout page discarded}%
228 \global\AtBegShi@Discardedfalse
229 \begingroup
230 \setbox\AtBeginShipoutBox\box\AtBeginShipoutBox
231 \endgroup
232 \let\protect\AtBegShi@OrgProtect
233 \else
234 \AtBegShi@First
235 \let\protect\AtBegShi@OrgProtect
236 \AtBeginShipoutOriginalShipout\box\AtBeginShipoutBox
237 \fi
238 \fi
239 }

240 \catcode'\X=11 %
241 \catcode'\E=11 %

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

```

```

251     }%
252     \wd\ltx@zero=Opt %
253     \ht\ltx@zero=Opt %
254     \dp\ltx@zero=Opt %
255     \global\setbox\AtBeginShipoutBox=\vbox{%
256         \baselineskip Opt\relax
257         \lineskip Opt\relax
258         \lineskiplimit Opt\relax
259         \copy\ltx@zero
260         \copy\AtBeginShipoutBox
261     }%
262     \fi
263     \global\let\AtBegShi@First\@empty
264     \global\let\AtBeginShipoutFirst\AtBegShi@FirstDisabled
265 \endgroup
266 }

\AtBegShi@Hook

267 \gdef\AtBegShi@Hook{}

\AtBegShi@HookNext

268 \gdef\AtBegShi@HookNext{}

\AtBegShi@HookFirst

269 \gdef\AtBegShi@HookFirst{}

\AtBeginShipout

270 \AtBegShi@CheckDefinable\AtBeginShipout
271 \def\AtBeginShipout{%
272     \AtBegShi@AddHook\AtBegShi@Hook
273 }

\AtBeginShipoutNext

274 \AtBegShi@CheckDefinable\AtBeginShipoutNext
275 \def\AtBeginShipoutNext{%
276     \AtBegShi@AddHook\AtBegShi@HookNext
277 }

\AtBeginShipoutFirst

278 \AtBegShi@CheckDefinable\AtBeginShipoutFirst
279 \def\AtBeginShipoutFirst{%
280     \AtBegShi@AddTo\AtBegShi@HookFirst
281 }

\AtBegShi@FirstDisabled

282 \long\def\AtBegShi@FirstDisabled#1{%
283     \@PackageWarning{atbegshi}{%
284         First page is already shipped out, ignoring\MessageBreak
285         \string\AtBeginShipoutFirst
286     }%
287 }

\AtBegShi@AddTo

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

```

```

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

\AtBegShi@AddHook

299 \long\def\AtBegShi@AddHook#1#2{%
300   \AtBegShi@AddTo#1{\AtBegShi@Item{#2}}%
301 }

\AtBegShi@Item

302 \long\def\AtBegShi@Item#1{%
303   \ifAtBegShi@Discarded
304   \else
305     #1%
306     \ifvoid\AtBeginShipoutBox
307       \@PackageWarning{atbegshi}{%
308         Shipout box was voided by hook,\MessageBreak
309         ignoring shipout box%
310       }%
311       \AtBeginShipoutDiscard
312     \fi
313   \fi
314 }

\AtBeginShipoutInit

315 \AtBegShi@CheckDefinable\AtBeginShipoutInit
316 \def\AtBeginShipoutInit{%
317   \ltx@ifundefined{newbox}{%
318     \@PackageError{atbegshi}{%
319       \string\AtBeginShipoutInit\space failed\MessageBreak
320       because of missing \expandafter\string\csname newbox\endcsname
321     }\@ehc
322   }{%
323     \csname newbox\endcsname\AtBeginShipoutBox
324     \AtBegShi@CheckDefinable\AtBeginShipoutOriginalShipout
325     \global\let\AtBeginShipoutOriginalShipout\shipout
326     \global\let\shipout\AtBegShi@Shipout
327   }%
328   \gdef\AtBeginShipoutInit{}%
329 }

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

```

### 3.4 Positioning

```

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

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

```

```

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

```



```

410 }%
411 \rlap{%
412   \kern-\AtBegShi@horigin\relax
413   \vbox to Opt{%
414     \kern-\AtBegShi@vorigin\relax
415     \kern-\ht\AtBeginShipoutBox
416     \AtBegShi@BeginPicture
417     #1%
418     \AtBegShi@endPicture
419     \vss
420   }%
421 }%
422 \hss
423 }%
424 }

```

### 3.5 Patches

Patches for L<sup>A</sup>T<sub>E</sub>X packages that redefine `\shipout`. L<sup>A</sup>T<sub>E</sub>X is now supposed to use  $\varepsilon$ -T<sub>E</sub>X. Thus we do not patch, without L<sup>A</sup>T<sub>E</sub>X and  $\varepsilon$ -T<sub>E</sub>X.

```

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

```

#### 3.5.1 Package crop

Fix of method and box.

```

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

```

```

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

```

### 3.5.2 Package everyshi

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

```

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

```

```

584 \fi
585 \endgroup
586 \let\AtBegShi@PatchEveryshi\relax
587 }
588 \@ifpackageloaded{everyshi}{%
589 \AtBegShi@PatchEveryshi
590 }{%
591 \AtBeginDocument{\AtBegShi@PatchEveryshi}%
592 }

```

### 3.5.3 Class memoir

Fix of method and box.

```

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

```

```

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

```

## 4 Test

### 4.1 Catcode checks for loading

```

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

```

```

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

```

```

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

768 </test1>

769 (*test2)
770 \input atbegshi.sty\relax
771 \def\msg#\{\immediate\write16}
772 \msg{File: atbegshi-test2.tex 2010/03/25 v1.12 Test file for plain-TeX}
773 \def\testmsg#1#2{%
774   \msg{}%
775   \msg{*** Test with box (#1), expected page output [#2]}}% hash-ok
776 }
777
778 \newbox\voidbox
779 \def\void{\box\voidbox}
780 \begingroup
781   \setbox\voidbox=\void
782 \endgroup
783
784 \count0=0\relax
785 \AtBeginShipout{%
786   \global\advance\count0 by 1\relax
787   \msg{* Inside \string\AtBeginShipout: [\the\count0]}}%
788 }
789
790 \AtBeginShipoutFirst{%
791   \msg{* Inside \string\AtBeginShipoutFirst}%
792   Hello World%
793 }
794
795 \testmsg{\string\null}{1}
796 \shipout\null
797
798 \AtBeginShipoutFirst{%
799   This is too late%
800 }
801
802 \testmsg{void}{}
803 \shipout\void
804
805 \testmsg{\string\copy255 (not void)}{2}
806 \setbox255\hbox{\vrule height 10bp width 10bp}
807 \shipout\copy255 %
808
809 \testmsg{\string\copy255 (again)}{3}
810 \shipout\copy255 %
811
812 \testmsg{\string\box255}{4}
813 \shipout\box255 %
814
815 \testmsg{\string\box255 (again)}{}
816 \shipout\box255 %
817
818 \testmsg{\string\hbox}{5}
819 \shipout\hbox{\vrule height 5bp width 20bp}
820
821 \testmsg{\string\vbox}{6}

```

```

822 \shipout\vbox{\hrule height 20bp width 5bp}
823
824 \testmsg{\string\null, voided by hook}{%
825 \def\VoidBox{%
826   \begingroup
827     \setbox\AtBeginShipoutBox=\box\AtBeginShipoutBox
828   \endgroup
829 }
830 \AtBeginShipout{\VoidBox}
831 \shipout\null
832 \def\VoidBox{}
833
834 \msg{*** \string\begingroup}
835 \begingroup
836   \testmsg{void}{}%
837   \shipout\void
838 \msg{*** \string\endgroup}
839 \endgroup
840
841 \msg{*** \string\begingroup}
842 \begingroup
843   \testmsg{void}{}%
844   \shipout\void
845   \testmsg{\string\null}{8}%
846   \shipout\null
847 \msg{*** \string\endgroup}
848 \endgroup
849
850 \testmsg{output routine}{9}
851 Hello World
852 \vfill
853 \eject
854
855 \testmsg{\string\null\space(discarded)}{9}
856 \AtBeginShipout{%
857   \msg{* Inside \string\AtBeginShipout: DISCARD}%
858   \AtBeginShipoutDiscard
859 }
860 \shipout\null
861
862 \end
863 </test2>
864 <*test3>
865 \NeedsTeXFormat{LaTeX2e}
866 \ProvidesFile{atbegshi-test3.tex}[2010/03/25 v1.12 Test file for LaTeX]
867 \RequirePackage{color}
868 \pagecolor{yellow}
869 \documentclass[a5paper,showtrims]{memoir}
870 \usepackage{atbegshi}
871 \AtBeginShipout{%
872   \setbox\AtBeginShipoutBox=\vbox{%
873     \vbox to 0pt{%
874       \kern-1.5in %
875       \hbox to 0pt{%
876         \kern-1.5in %
877         \color{blue}%
878         \rule{1in}{1in}%
879       \hss
880     }%
881     \vss
882   }%
883   \hrule

```



```

884 \hbox{\vrule\box\AtBeginShipoutBox\vrule}%
885 \hrule
886 }%
887 }
888 \usepackage{eso-pic}
889 \makeatletter
890 \ifundefined{EveryShipout@Init}{%
891 \typeout{Test skipped}%
892 \@@end
893 }{}
894 \EveryShipout@Init
895 \let\EveryShipout@Init\relax
896 \makeatother
897 \AddToShipoutPicture{%
898 \hspace{.52\paperwidth}%
899 \colorbox{cyan}{%
900 \rule{0mm}{\paperheight}%
901 \hspace{.48\paperwidth}%
902 }%
903 }

```

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

```

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

```

## 5 Installation

### 5.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

[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  $\text{\TeX}$  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
```

---

<sup>1</sup><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):

```
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
```

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<sup>A</sup>T<sub>E</sub>X.** The `.dtx` chooses its action depending on the format:

**plain T<sub>E</sub>X:** Run `docstrip` and extract the files.

**L<sup>A</sup>T<sub>E</sub>X:** Generate the documentation.

If you insist on using L<sup>A</sup>T<sub>E</sub>X for `docstrip` (really, `docstrip` does not need L<sup>A</sup>T<sub>E</sub>X), 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<sup>A</sup>T<sub>E</sub>X:

```
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  $\epsilon$ -T<sub>E</sub>X 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`.

## 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.

## Symbols

\#	694, 756	\_	760
\%	759		
\@	695, 752	A	
\@@end	892	\AddToShipoutPicture	897
\@EveryShipout@Init	894, 895	\advance	733, 741, 786
\@EveryShipout@Org@Shipout		\afterassignment	
	468, 469, 642, 643		202, 439, 482, 527, 564, 596, 650
\@EveryShipout@Output	532, 534, 578	\aftergroup	81,
\@EveryShipout@Shipout	554, 555		209, 444, 488, 532, 570, 601, 656
\@EveryShipout@Test	527, 537, 564, 567	\AtBeginDocument	334, 522, 591, 687
\@PackageError	318	\AtBeginShipout	2, 6,
\@PackageInfoNoLine	227, 510, 581, 676		44, 270, 785, 787, 830, 856, 857, 871
\@PackageWarning	217, 283, 307	\AtBeginShipoutBox	45, 47, 199, 204,
\@cclv	440, 443, 453, 528,		216, 230, 236, 255, 260, 306,
	531, 597, 600, 608, 612, 613, 617		323, 391, 396, 403, 407, 409,
\@ehc	321		415, 483, 497, 500, 502, 507,
\@empty	244, 245, 263		565, 575, 577, 651, 661, 663,
\@firstofone	703, 706		665, 666, 668, 673, 827, 872, 884
\@gobble	700, 708	\AtBeginShipoutDiscard	
\@ifclassloaded	684		3, 29, 184, 311, 858
\@ifdefinable	180	\AtBeginShipoutFirst	
\@ifpackageloaded	519, 588		3, 264, 278, 285, 790, 791, 798
\@ifundefined	890	\AtBeginShipoutInit	3, 315, 332, 334
\@undefined	107, 165	\AtBeginShipoutNext	2, 14, 28, 274
\[	757	\AtBeginShipoutOriginalShipout	
\\	753		3, 236, 324,
\{	692, 754		325, 466, 467, 541, 542, 634, 635
\}	693, 755	\AtBeginShipoutUpperLeft	3, 7, 15, 390
\]	758	\AtBeginShipoutUpperLeftForeground	
			3, 406

<code>\AtBegShi@AbortIfUndefined</code> . . . . .	124, 125, 126, 130, 131, 132,
425, 432, 433, 434, 435	133, 137, 139, 191, 192, 194,
<code>\AtBegShi@AddHook</code> . . . . .	195, 240, 241, 692, 693, 694,
<code>\AtBegShi@AddTo</code> . . . . .	695, 730, 739, 752, 753, 754,
<code>\AtBegShi@AtEnd</code> . . . . .	755, 756, 757, 758, 759, 760, 761
<code>\AtBegShi@BeginPicture</code> . . . . .	<code>\circle</code> . . . . . 8
361, 377, 397, 416	<code>\color</code> . . . . . 16, 877
<code>\AtBegShi@CheckDefinable</code> . . . . .	<code>\colorbox</code> . . . . . 899
. 159, 184, 270, 274, 278, 315, 324	<code>\copy</code> . . . . . 259,
<code>\AtBegShi@Crop@@ship</code> . . . . .	260, 387, 409, 805, 807, 809, 810
<code>\AtBegShi@Crop@ship</code> . . . . .	<code>\count</code> . . . . . 784, 786, 787
<code>\AtBegShi@Crop@shiplist</code> . . . . .	<code>\count@</code> . . . . . 697, 726,
<code>\AtBegShi@Crop@shipout</code> . . . . .	730, 732, 733, 737, 739, 740, 741
. 438, 464, 466, 468, 470, 472, 474	<code>\countdef</code> . . . . . 697
<code>\AtBegShi@Discardedfalse</code> . . . . .	<code>\CROP@@ship</code> . . . . . 446, 462, 490, 499
<code>\AtBegShi@Discardedtrue</code> . . . . .	<code>\CROP@kernel</code> . . . . . 452, 496
<code>\AtBegShi@EndPicture</code> 371, 382, 399, 418	<code>\CROP@ship</code> . . . . . 439, 460, 482, 485
<code>\AtBegShi@Everyshi@Output</code> . . . . .	<code>\CROP@shiplist</code> . . . . . 457, 461, 492, 503
<code>\AtBegShi@Everyshi@shipout</code> . . . . .	<code>\CROP@shipout</code> . . . . .
539, 541, 543, 545, 547, 549, 554	. . . . . 456, 505, 543, 544, 636, 637
<code>\AtBegShi@Everyshi@Test</code> . . . . .	<code>\csname</code> . . . . . 65,
<code>\AtBegShi@First</code> . . . . .	73, 99, 115, 122, 152, 160, 190,
<code>\AtBegShi@FirstDisabled</code> . . . . .	220, 289, 320, 323, 331, 337,
<code>\AtBegShi@found</code> . . . . .	351, 354, 365, 427, 552, 696,
469, 471, 473, 475, 477, 480,	699, 702, 705, 744, 766, 904, 908
538, 540, 542, 544, 546, 548,	<code>\currentgrouplevel</code> . . . . . 201,
550, 555, 559, 562, 631, 633,	208, 481, 486, 563, 568, 649, 654
635, 637, 639, 641, 643, 645, 648	
<code>\AtBegShi@GroupLevel</code> . . . . .	
208, 481, 486, 563, 568, 649, 654	<b>D</b>
<code>\AtBegShi@Hook</code> . . . . .	<code>\deadcycles</code> . . . . . 186
<code>\AtBegShi@HookFirst</code> 245, 249, 269, 280	<code>\documentclass</code> . . . . . 2, 37, 869
<code>\AtBegShi@HookNext</code> . . . . .	<code>\dp</code> . . . . . 254, 386
<code>\AtBegShi@horigin</code> . . . . .	
<code>\AtBegShi@Item</code> . . . . .	<b>E</b>
<code>\AtBegShi@Memoir@shipi</code> . . . . .	<code>\E</code> . . . . . 241
<code>\AtBegShi@Memoir@shipiiA</code> . . . . .	<code>\eject</code> . . . . . 853
<code>\AtBegShi@Memoir@shipiiB</code> . . . . .	<code>\empty</code> . . . . . 68, 69
<code>\AtBegShi@Memoir@shipout</code> . . . . .	<code>\end</code> . . . . . 34, 54, 767, 862, 910
. 595, 632, 634, 636, 638, 640, 642	<code>\endcsname</code> . . . . . 65,
<code>\AtBegShi@OrgProtect</code> . . . . .	73, 99, 115, 122, 152, 160, 190,
<code>\AtBegShi@Output</code> . . . . .	220, 289, 320, 323, 331, 337,
<code>\AtBegShi@PatchCrop</code> 436, 517, 520, 522	351, 354, 365, 427, 552, 696,
<code>\AtBegShi@PatchEveryshi</code> . . . . .	699, 702, 705, 744, 766, 904, 908
. . . . . 524, 586, 589, 591	<code>\endinput</code> . . . . . 81, 429
<code>\AtBegShi@PatchMemoir</code> . . . . .	<code>\endpicture</code> . . . . . 372
. . . . . 593, 682, 685, 687	<code>\errmessage</code> . . . . . 171
<code>\AtBegShi@Shipout</code> . . . . .	
<code>\AtBegShi@Test</code> . . . . .	<b>F</b>
<code>\AtBegShi@vorigin</code> . . . . .	<code>\fill</code> . . . . . 25
<b>B</b>	<b>G</b>
<code>\baselineskip</code> . . . . .	<code>\g@addto@macro</code> . . . . . 297
<code>\begin</code> . . . . .	<code>\gdef</code> . . . . . 223, 267, 268, 269, 328
<code>\body</code> . . . . .	<code>\GPTorg@shipout</code> . . . . .
<code>\box</code> . . . . .	. . . . . 470, 471, 545, 546, 638, 639
47, 230, 236, 403, 453, 497,	
506, 577, 613, 668, 672, 779,	<b>H</b>
812, 813, 815, 816, 827, 884, 908	<code>\hbox</code> . . . . .
	45, 199, 378,
<b>C</b>	391, 407, 806, 818, 819, 875, 884
<code>\catcode</code> . . . . .	<code>\hrule</code> . . . . . 822, 883, 885
58, 59, 60, 61, 62, 63, 64,	<code>\hspace</code> . . . . . 898, 901
72, 86, 87, 88, 89, 90, 91, 92, 93,	<code>\hss</code> . . . . . 422, 879
94, 95, 96, 97, 98, 119, 120, 123,	<code>\ht</code> . . . . . 253, 385, 396, 415

<b>I</b>	
<code>\ifAtBegShi@Discarded</code>	183, 226, 303
<code>\ifcase</code>	162, 350, 622
<code>\ifdim</code>	207
<code>\ifnum</code>	208, 486, 568, 654, 732, 740
<code>\ifpdf</code>	342
<code>\ifvbox</code>	665
<code>\ifvoid</code>	216, 306, 443, 500, 531, 575, 600, 612, 661
<code>\ifx</code>	66, 69, 73, 99, 107, 110, 152, 160, 162, 165, 190, 245, 289, 331, 337, 351, 354, 365, 427, 460, 461, 462, 464, 466, 468, 470, 472, 474, 477, 537, 539, 541, 543, 545, 547, 549, 552, 554, 559, 621, 622, 625, 632, 634, 636, 638, 640, 642, 645, 696, 699, 702, 705, 744
<code>\ignorespaces</code>	369, 380
<code>\immediate</code>	75, 101, 771
<code>\input</code>	153, 154, 338, 745, 770
<code>\iterate</code>	713, 715, 717
<b>K</b>	
<code>\kern</code>	200, 393, 395, 396, 412, 414, 415, 874, 876
<b>L</b>	
<code>\lastkern</code>	207
<code>\line</code>	17, 18
<code>\lineskip</code>	257, 449, 493
<code>\lineskiplimit</code>	258, 450, 494
<code>\LoadCommand</code>	745, 762
<code>\loop</code>	711, 727, 738
<code>\ltx@cclv</code>	577
<code>\ltx@ifUndefined</code>	317
<code>\ltx@newif</code>	183
<code>\ltx@one</code>	163, 166, 628
<code>\ltx@zero</code>	168, 186, 247, 252, 253, 254, 259, 292, 293, 378, 385, 386, 387, 623, 626
<b>M</b>	
<code>\makeatletter</code>	889
<code>\makeatother</code>	896
<code>\mem@oldshipout</code>	474, 475, 549, 550, 606, 613, 615, 671
<code>\mem@shipi</code>	596, 621, 650, 653
<code>\mem@shipii</code>	603, 622, 625, 658, 660
<code>\MessageBreak</code>	284, 308, 319
<code>\msg</code>	771, 772, 774, 775, 787, 791, 834, 838, 841, 847, 857
<b>N</b>	
<code>\NeedsTeXFormat</code>	865
<code>\newbox</code>	778
<code>\newpage</code>	13, 22, 27, 32, 52
<code>\next</code>	717, 719, 721
<code>\null</code>	795, 796, 824, 831, 845, 846, 855, 860, 907
<code>\number</code>	201, 481, 563, 649
<b>P</b>	
<code>\p@</code>	200
<code>\PackageInfo</code>	78
<code>\pagecolor</code>	868
<code>\paperheight</code>	8, 17, 18, 900
<code>\paperwidth</code>	8, 17, 18, 898, 901
<code>\par</code>	24
<code>\pdfhorigin</code>	343
<code>\pdfvorigin</code>	344
<code>\picture</code>	363
<code>\protect</code>	219, 232, 235
<code>\ProvidesFile</code>	866
<code>\ProvidesPackage</code>	70, 116
<code>\put</code>	8, 17, 18
<b>R</b>	
<code>\RangeCatcodeInvalid</code>	736, 748, 749, 750, 751
<code>\repeat</code>	711, 723, 734, 742
<code>\RequirePackage</code>	156, 157, 340, 867
<code>\RestoreCatcodes</code>	725, 728, 729, 763
<code>\rlap</code>	392, 408, 411
<code>\rule</code>	878, 900
<b>S</b>	
<code>\section</code>	12, 909
<code>\setbox</code>	45, 199, 204, 230, 247, 255, 378, 391, 407, 440, 483, 502, 528, 565, 577, 597, 651, 663, 781, 806, 827, 872
<code>\shipout</code>	325, 326, 464, 465, 539, 540, 632, 633, 796, 803, 807, 810, 813, 816, 819, 822, 831, 837, 844, 846, 860, 907, 908
<code>\space</code>	172, 319, 855
<code>\special</code>	46
<b>T</b>	
<code>\Test</code>	747, 765
<code>\testmsg</code>	773, 795, 802, 805, 809, 812, 815, 818, 821, 824, 836, 843, 845, 850, 855
<code>\THBorg@shipout</code>	472, 473, 547, 548, 640, 641
<code>\the</code>	123, 124, 125, 126, 137, 293, 730, 787
<code>\TMP@EnsureCode</code>	134, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150
<code>\toks</code>	292, 293
<code>\trimmarks</code>	607, 616, 664
<code>\typeout</code>	891
<b>U</b>	
<code>\unitlength</code>	367
<code>\unvbox</code>	608, 617, 666
<code>\usepackage</code>	3, 4, 5, 38, 39, 870, 888, 905
<b>V</b>	
<code>\vbox</code>	247, 255, 394, 413, 456, 502, 606, 615, 663, 821, 822, 872, 873
<code>\vfill</code>	852
<code>\void</code>	779, 781, 803, 837, 844
<code>\VoidBox</code>	825, 830, 832
<code>\voidbox</code>	778, 779, 781
<code>\vrule</code>	806, 819, 884
<code>\vspace</code>	25
<code>\vss</code>	400, 419, 881

<b>W</b>			
<code>\wd</code>	.....	<code>\x</code>	.....
<code>\write</code>	.....		.....
			.....
<b>X</b>		<b>Z</b>	
<code>\X</code>	.....	<code>\z@</code>	.....