

# The atbegshi package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/09/09 v1.6

## Abstract

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

## Contents

<b>1</b>	<b>Documentation</b>	<b>2</b>
1.1	Example . . . . .	3
<b>2</b>	<b>Method of <code>\shipout</code> overloading</b>	<b>4</b>
2.1	<code>\shipout</code> . . . . .	4
2.2	<code>\afterassignment</code> . . . . .	4
2.3	Test for direct or indirect boxes . . . . .	5
2.3.1	With $\varepsilon$ -TeX . . . . .	5
2.3.2	Without $\varepsilon$ -TeX . . . . .	5
2.3.3	<code>\lastkern</code> method . . . . .	6
2.4	Output . . . . .	7
2.5	Separate box register . . . . .	7
2.6	Summary . . . . .	7
2.6.1	With $\varepsilon$ -TeX . . . . .	7
2.6.2	Without $\varepsilon$ -TeX, traditional way . . . . .	8
2.6.3	<code>\lastkern</code> method . . . . .	9
<b>3</b>	<b>Implementation</b>	<b>9</b>
3.1	Reload check and package identification . . . . .	9
3.2	Catcodes . . . . .	10
3.3	Preparations . . . . .	10
3.4	Positioning . . . . .	14
3.5	Patches . . . . .	15
3.5.1	Package <code>crop</code> . . . . .	15
3.5.2	Package <code>everyshi</code> . . . . .	17
3.5.3	Class <code>memoir</code> . . . . .	18
<b>4</b>	<b>Test</b>	<b>20</b>
4.1	Catcode checks for loading . . . . .	20
<b>5</b>	<b>Installation</b>	<b>23</b>
5.1	Download . . . . .	23
5.2	Bundle installation . . . . .	23
5.3	Package installation . . . . .	23
5.4	Refresh file name databases . . . . .	24
5.5	Some details for the interested . . . . .	24

<b>6 History</b>	<b>24</b>
[2007/04/17 v1.0]	24
[2007/04/18 v1.1]	25
[2007/04/19 v1.2]	25
[2007/04/26 v1.3]	25
[2007/04/27 v1.4]	25
[2007/06/06 v1.5]	25
[2007/09/09 v1.6]	25
<b>7 Index</b>	<b>25</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 pdf<sub>T</sub>E<sub>X</sub> in PDF mode, the settings of `\pdfhorigin` and `\pdfvorigin` are respected.

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}
```

## 1.1 Example

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

```
1 <*example>
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 \example

```

## 2 Method of \shipout overloading

### 2.1 \shipout

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

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

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

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

Then we have to differentiate between void and empty boxes:

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

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

### 2.2 \afterassignment

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

```

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

```

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

```

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

```

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

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

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

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

```
\shipout\box0 ⇒ \setbox\mybox=\box0 \@test
```

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

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

## 2.3 Test for direct or indirect boxes

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

### 2.3.1 With $\varepsilon$ -TeX

With the  $\varepsilon$ -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 $\varepsilon$ -TeX

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

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

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

Unhappily all these qualities even combined are not sufficient for constructing an initial box value, because `\shipout` can be called with a box that is accidentally just the same as the chosen 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 0pt. 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 0pt.

```
\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
\begin{group}
\setbox\mybox=\box\mybox % ensure \mybox is void
\end{group}
\let\original@shipout\shipout

\shipout :=
% trick to get a void box \mybox
\begin{group}
\setbox\mybox=\box\mybox
\end{group}
\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  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  without  $\varepsilon\text{-}\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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  $\varepsilon\text{-}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 's `\currentgrouplevel`, if it is available. Otherwise the `\lastkern` method is used.

```
36 <*package>
```

### 3.1 Reload check and package identification

Reload check, especially if the package is not used with  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ .

```
37 \begingroup
38 \catcode44 12 % ,
39 \catcode45 12 % -
40 \catcode46 12 % .
41 \catcode58 12 % :
42 \catcode64 11 % @
43 \expandafter\let\expandafter\x\csname ver@atbegshi.sty\endcsname
44 \ifcase 0%
45   \ifx\x\relax % plain
46   \else
47     \ifx\x\empty % LaTeX
48     \else
49       1%
50     \fi
51   \fi
52 \else
53   \expandafter\ifx\csname PackageInfo\endcsname\relax
54     \def\x#1#2{%
```

```

55     \immediate\write-1{Package #1 Info: #2.}%
56   }%
57   \else
58     \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
59   \fi
60   \x{atbegshi}{The package is already loaded}%
61   \endgroup
62   \expandafter\endinput
63 \fi
64 \endgroup
Package identification:
65 \begingroup
66   \catcode40 12 % (
67   \catcode41 12 % )
68   \catcode44 12 % ,
69   \catcode45 12 % -
70   \catcode46 12 % .
71   \catcode47 12 % /
72   \catcode58 12 % :
73   \catcode64 11 % @
74   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
75     \def\x#1#2#3[#4]{\endgroup
76       \immediate\write-1{Package: #3 #4}%
77       \xdef#1{#4}%
78     }%
79   \else
80     \def\x#1#2[#3]{\endgroup
81       #2[{#3}]%
82       \ifx#1\relax
83         \xdef#1{#3}%
84       \fi
85     }%
86   \fi
87   \expandafter\x\csname ver@atbegshi.sty\endcsname
88   \ProvidesPackage{atbegshi}%
89   [2007/09/09 v1.6 At begin shipout hook (H0)]

```

### 3.2 Catcodes

```

90 \expandafter\edef\csname AtBegShi@AtEnd\endcsname{%
91   \catcode64 \the\catcode64\relax
92 }
93 \catcode64 11 % @
94 \def\TMP@EnsureCode#1#2{%
95   \edef\AtBegShi@AtEnd{%
96     \AtBegShi@AtEnd
97     \catcode#1 \the\catcode#1\relax
98   }%
99   \catcode#1 #2\relax
100 }
101 \TMP@EnsureCode{40}{12}% (
102 \TMP@EnsureCode{41}{12}% )
103 \TMP@EnsureCode{44}{12}% ,
104 \TMP@EnsureCode{45}{12}% -
105 \TMP@EnsureCode{47}{12}% /
106 \TMP@EnsureCode{46}{12}% .
107 \TMP@EnsureCode{58}{12}% :
108 \TMP@EnsureCode{61}{12}% =
109 \TMP@EnsureCode{94}{7}% ^ (superscript)
110 \TMP@EnsureCode{96}{12}% ‘

```

### 3.3 Preparations

```

111 \begingroup\expandafter\expandafter\expandafter\endgroup
112 \expandafter\ifx\csname RequirePackage\endcsname\relax
113   \input infwarerr.sty\relax
114 \else
115   \RequirePackage{infwarerr}[2007/09/09]%
116 \fi

```

\AtBegShi@CheckDefinable

```

117 \begingroup\expandafter\expandafter\expandafter\endgroup
118 \expandafter\ifx\csname @ifdefinable\endcsname\relax
119   \def\AtBegShi@CheckDefinable#1{%
120     \ifcase\ifx#1\relax
121       \@ne
122     \else
123       \ifx#1\@undefined
124         \@ne
125       \else
126         \z@
127       \fi
128     \fi
129     \errmessage{%
130       Package atbegshi: \string#1\space
131       is already defined%
132     }%
133   \endgroup
134 \fi
135 }%
136 \else
137   \def\AtBegShi@CheckDefinable#1{%
138     \@ifdefinable{#1}{}%
139   }%
140 \fi

141 \newif\ifAtBegShi@Discarded

```

\AtBeginShipoutDiscard

```

142 \AtBegShi@CheckDefinable\AtBeginShipoutDiscard
143 \def\AtBeginShipoutDiscard{%
144   \deadcycles=\z@
145   \global\AtBegShi@Discardedtrue
146 }

147 \begingroup\expandafter\expandafter\expandafter\endgroup
148 \expandafter\ifx\csname currentgrouplevel\endcsname\relax
149   \catcode'X=9 % ignore
150   \catcode'E=14 % comment
151 \else
152   \catcode'X=14 % comment
153   \catcode'E=9 % ignore
154 \fi

```

\AtBegShi@Shipout

```

155 \def\AtBegShi@Shipout{%
156 X \begingroup
157 X \setbox\AtBeginShipoutBox=\hbox\bgroup
158 X \kern\p@
159 E \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
160 \afterassignment\AtBegShi@Test
161 X \global
162 \setbox\AtBeginShipoutBox=%
163 }

```

\AtBegShi@Test

```
164 \def\AtBegShi@Test{%
165 X \ifdim\lastkern=\z@
166 E \ifnum\AtBegShi@GroupLevel<\currentgrouplevel
167 \expandafter\aftergroup
168 \fi
169 \AtBegShi@Output
170 }
```

\AtBegShi@Output

```
171 \def\AtBegShi@Output{%
172 X \egroup
173 X \endgroup
174 \ifvoid\AtBeginShipoutBox
175 \@PackageWarning{atbegshi}{Ignoring void shipout box}%
176 \else
177 \let\AtBegShi@OrgProtect\protect
178 \csname set@typeset@protect\endcsname
179 \global\AtBegShi@Discardedfalse
180 \AtBegShi@Hook
181 \AtBegShi@HookNext
182 \gdef\AtBegShi@HookNext{}%
183 \ifAtBegShi@Discarded
184 \@PackageInfoNoLine{atbegshi}{Shipout page discarded}%
185 \global\AtBegShi@Discardedfalse
186 \begingroup
187 \setbox\AtBeginShipoutBox\box\AtBeginShipoutBox
188 \endgroup
189 \let\protect\AtBegShi@OrgProtect
190 \else
191 \AtBegShi@First
192 \let\protect\AtBegShi@OrgProtect
193 \AtBegShi@OrgShipout\box\AtBeginShipoutBox
194 \fi
195 \fi
196 }

197 \catcode'\X=11 %
198 \catcode'\E=11 %
```

\AtBegShi@First

```
199 \def\AtBegShi@First{%
200 \begingroup
201 \def\@empty{}%
202 \ifx\AtBegShi@HookFirst\@empty
203 \else
204 \setbox\z@=\vbox{%
205 \begingroup
206 \AtBegShi@HookFirst
207 \endgroup
208 }%
209 \wd\z@=\z@
210 \ht\z@=\z@
211 \dp\z@=\z@
212 \global\setbox\AtBeginShipoutBox=\vbox{%
213 \baselineskip\z@skip
214 \lineskip\z@skip
215 \lineskiplimit\z@
216 \copy\z@
217 \copy\AtBeginShipoutBox
218 }%
219 \fi
```

```

220     \global\let\AtBegShi@First\@empty
221     \global\let\AtBeginShipoutFirst\AtBegShi@FirstDisabled
222 \endgroup
223 }

\AtBegShi@Hook
224 \gdef\AtBegShi@Hook{}

\AtBegShi@HookNext
225 \gdef\AtBegShi@HookNext{}

\AtBegShi@HookFirst
226 \gdef\AtBegShi@HookFirst{}

\AtBeginShipout
227 \AtBegShi@CheckDefinable\AtBeginShipout
228 \def\AtBeginShipout{%
229     \AtBegShi@AddHook\AtBegShi@Hook
230 }

\AtBeginShipoutNext
231 \AtBegShi@CheckDefinable\AtBeginShipoutNext
232 \def\AtBeginShipoutNext{%
233     \AtBegShi@AddHook\AtBegShi@HookNext
234 }

\AtBeginShipoutFirst
235 \AtBegShi@CheckDefinable\AtBeginShipoutFirst
236 \def\AtBeginShipoutFirst{%
237     \AtBegShi@AddTo\AtBegShi@HookFirst
238 }

\AtBegShi@FirstDisabled
239 \long\def\AtBegShi@FirstDisabled#1{%
240     \@PackageWarning{atbegshi}{%
241         First page is already shipped out, ignoring\MessageBreak
242         \string\AtBeginShipoutFirst
243     }%
244 }

\AtBegShi@AddTo
245 \begingroup\expandafter\expandafter\expandafter\endgroup
246 \expandafter\ifx\csname g@addto@macro\endcsname\relax
247     \long\def\AtBegShi@AddTo#1#2{%
248         \begingroup
249             \toks\z@\expandafter{#1#2}%
250             \xdef#1{\the\toks\z@}%
251         \endgroup
252     }%
253 \else
254     \let\AtBegShi@AddTo\g@addto@macro
255 \fi

\AtBegShi@AddHook
256 \long\def\AtBegShi@AddHook#1#2{%
257     \AtBegShi@AddTo#1{\AtBegShi@Item{#2}}%
258 }

```

\AtBegShi@Item

```
259 \long\def\AtBegShi@Item#1{%
260   \ifAtBegShi@Discarded
261   \else
262     #1%
263   \ifvoid\AtBeginShipoutBox
264     \@PackageWarning{atbegshi}{%
265       Shipout box was voided by hook,\MessageBreak
266       ignoring shipout box%
267     }%
268     \AtBeginShipoutDiscard
269   \fi
270 \fi
271 }
```

\AtBeginShipoutInit

```
272 \AtBegShi@CheckDefinable\AtBeginShipoutInit
273 \def\AtBeginShipoutInit{%
274   \csname newbox\endcsname\AtBeginShipoutBox
275   \AtBegShi@CheckDefinable\AtBegShi@OrgShipout
276   \global\let\AtBegShi@OrgShipout\shipout
277   \global\let\shipout\AtBegShi@Shipout
278   \gdef\AtBeginShipoutInit{}%
279 }

280 \begingroup\expandafter\expandafter\expandafter\endgroup
281 \expandafter\ifx\csname AtBeginDocument\endcsname\relax
282   \AtBeginShipoutInit
283 \else
284   \AtBeginDocument{\AtBeginShipoutInit}%
285 \fi
```

### 3.4 Positioning

```
286 \begingroup\expandafter\expandafter\expandafter\endgroup
287 \expandafter\ifx\csname RequirePackage\endcsname\relax
288   \input ifpdf.sty\relax
289 \else
290   \RequirePackage{ifpdf}\relax
291 \fi

292 \ifpdf
293   \def\AtBegShi@horigin{\pdfhorigin}%
294   \def\AtBegShi@vorigin{\pdfvorigin}%
295 \else
296   \def\AtBegShi@horigin{72.27pt}%
297   \def\AtBegShi@vorigin{72.27pt}%
298 \fi

299 \begingroup
300 \ifcase
301   \expandafter\ifx\csname picture\endcsname\relax
302     1%
303   \else
304     \expandafter\ifx\csname endpicture\endcsname\relax
305       1%
306     \else
307       0%
308     \fi
309   \fi
310 \endgroup
311 \def\AtBegShi@BeginPicture{%
312   \begingroup
```

```

313     \picture(0,0)\relax
314     \begingroup\expandafter\expandafter\expandafter\endgroup
315     \expandafter\ifx\csname unitlength\endcsname\relax
316     \else
317         \unitlength=1pt\relax
318     \fi
319     \ignorespaces
320 }%
321 \def\AtBegShi@EndPicture{%
322     \endpicture
323     \endgroup
324 }%
325 \else
326     \endgroup
327     \def\AtBegShi@BeginPicture{%
328         \setbox0=\hbox\bgroup
329         \begingroup
330         \ignorespaces
331     }%
332     \def\AtBegShi@EndPicture{%
333         \endgroup
334         \egroup
335         \ht0=0pt\relax
336         \dp0=0pt\relax
337         \copy0 %
338     }%
339 \fi
340 \def\AtBeginShipoutUpperLeft#1{%
341     \global\setbox\AtBeginShipoutBox=\hbox{%
342         \rlap{%
343             \kern-\AtBegShi@horigin\relax
344             \vbox to 0pt{%
345                 \kern-\AtBegShi@vorigin\relax
346                 \kern-\ht\AtBeginShipoutBox
347                 \AtBegShi@BeginPicture
348                 #1%
349                 \AtBegShi@EndPicture
350                 \vss
351             }%
352         }%
353         \box\AtBeginShipoutBox
354     }%
355 }

```

### 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.

```

356 \def\AtBegShi@AbortIfUndefined#1{%
357     \begingroup\expandafter\expandafter\expandafter\endgroup
358     \expandafter\ifx\csname#1\endcsname\relax
359         \AtBegShi@AtEnd
360     \expandafter\endinput
361 \fi
362 }
363 \AtBegShi@AbortIfUndefined{currentgrouplevel}
364 \AtBegShi@AbortIfUndefined{AtBeginDocument}
365 \AtBegShi@AbortIfUndefined{@ifpackageloaded}
366 \AtBegShi@AbortIfUndefined{@ifclassloaded}

```

#### 3.5.1 Package crop

Fix of method and box.

```

367 \def\AtBegShi@PatchCrop{%
368   \begingroup
369   \def\AtBegShi@Crop@shipout{%
370     \afterassignment\CROP@ship
371     \setbox\@cclv=%
372   }%
373   \def\AtBegShi@Crop@ship{%
374     \ifvoid\@cclv
375       \expandafter\aftergroup
376       \fi
377     \CROP@@ship
378   }%
379   \def\AtBegShi@Crop@shiplist{%
380     \lineskip\z@
381     \lineskiplimit\z@
382     \baselineskip\z@
383     \CROP@kernel
384     \box\@cclv
385   }%
386   \def\AtBegShi@Crop@@ship{%
387     \CROP@shipout\vbox{%
388       \CROP@shiplist
389     }%
390   }%
391   \ifx\AtBegShi@Crop@ship\CROP@ship
392     \ifx\AtBegShi@Crop@shiplist\CROP@shiplist
393       \ifx\AtBegShi@Crop@@ship\CROP@@ship
394         \let\AtBegShi@found\relax
395         \ifx\shipout\AtBegShi@Crop@shipout
396           \def\AtBegShi@found{\shipout}%
397         \else\ifx\AtBegShi@OrgShipout\AtBegShi@Crop@shipout
398           \def\AtBegShi@found{\AtBegShi@OrgShipout}%
399         \else\ifx\@EveryShipout@Org@Shipout\AtBegShi@Crop@shipout
400           \def\AtBegShi@found{\@EveryShipout@Org@Shipout}%
401         \else\ifx\GPTorg@shipout\AtBegShi@Crop@shipout
402           \def\AtBegShi@found{\GPTorg@shipout}%
403         \else\ifx\THBorg@shipout\AtBegShi@Crop@shipout
404           \def\AtBegShi@found{\THBorg@shipout}%
405         \else\ifx\mem@oldshipout\AtBegShi@Crop@shipout
406           \def\AtBegShi@found{\mem@oldshipout}%
407         \fi\fi\fi\fi\fi\fi
408         \ifx\AtBegShi@found\relax
409           \else
410             \expandafter\endgroup
411             \expandafter\def\AtBegShi@found{%
412               \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
413               \afterassignment\CROP@ship
414               \setbox\AtBeginShipoutBox=%
415             }%
416             \def\CROP@ship{%
417               \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
418               \else
419                 \expandafter\aftergroup
420                 \fi
421               \CROP@@ship
422             }%
423             \def\CROP@shiplist{%
424               \lineskip\z@
425               \lineskiplimit\z@
426               \baselineskip\z@
427               \CROP@kernel
428               \box\AtBeginShipoutBox

```



```

429         }%
430         \def\CROP@ship{%
431             \ifvoid\AtBeginShipoutBox
432             \else
433                 \setbox\AtBeginShipoutBox=\vbox{%
434                     \CROP@shiplist
435                 }%
436                 \expandafter\CROP@shipout
437                 \expandafter\box
438                 \expandafter\AtBeginShipoutBox
439             \fi
440         }%
441         \@PackageInfoNoLine{atbegshi}{Package 'crop' patched}%
442         \begingroup
443         \fi
444     \fi
445 \fi
446 \fi
447 \endgroup
448 \let\AtBegShi@PatchCrop\relax
449 }
450 \ifpackageloaded{crop}{%
451     \AtBegShi@PatchCrop
452 }{%
453     \AtBeginDocument{\AtBegShi@PatchCrop}%
454 }

```

### 3.5.2 Package everyshi

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

```

455 \def\AtBegShi@PatchEveryshi{%
456     \begingroup
457     \long\def\AtBegShi@Everyshi@shipout{%
458         \afterassignment\@EveryShipout@Test
459         \global\setbox\@cclv= %
460     }%
461     \long\def\AtBegShi@Everyshi@Test{%
462         \ifvoid\@cclv\relax
463             \aftergroup\@EveryShipout@Output
464         \else
465             \@EveryShipout@Output
466         \fi
467     }%
468     \ifx\AtBegShi@Everyshi@Test\@EveryShipout@Test
469         \let\AtBegShi@found\relax
470         \ifx\shipout\AtBegShi@Everyshi@shipout
471             \def\AtBegShi@found{\shipout}%
472         \else\ifx\AtBegShi@OrgShipout\AtBegShi@Everyshi@shipout
473             \def\AtBegShi@found{\AtBegShi@OrgShipout}%
474         \else\ifx\CROP@shipout\AtBegShi@Everyshi@shipout
475             \def\AtBegShi@found{\CROP@shipout}%
476         \else\ifx\GPTorg@shipout\AtBegShi@Everyshi@shipout
477             \def\AtBegShi@found{\GPTorg@shipout}%
478         \else\ifx\THBorg@shipout\AtBegShi@Everyshi@shipout
479             \def\AtBegShi@found{\THBorg@shipout}%
480         \else\ifx\mem@oldshipout\AtBegShi@Everyshi@shipout
481             \def\AtBegShi@found{\mem@oldshipout}%
482         \else
483             \expandafter\ifx\csname @EveryShipout@Org@Shipout\endcsname
484                 \relax
485             \ifx\@EveryShipout@Shipout\AtBegShi@Everyshi@shipout
486                 \def\AtBegShi@found{\@EveryShipout@Shipout}%

```

```

487         \fi
488     \fi
489     \fi\fi\fi\fi\fi\fi
490     \ifx\AtBegShi@found\relax
491     \else
492         \expandafter\endgroup
493         \expandafter\def\AtBegShi@found{%
494             \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
495             \afterassignment\@EveryShipout@Test
496             \setbox\AtBeginShipoutBox=%
497             }%
498         \def\@EveryShipout@Test{%
499             \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
500             \else
501                 \expandafter\aftergroup
502                 \fi
503             \AtBegShi@Everyshi@Output
504             }%
505         \def\AtBegShi@Everyshi@Output{%
506             \ifvoid\AtBeginShipoutBox
507             \else
508                 \global\setbox\@cclv\box\AtBeginShipoutBox
509                 \expandafter\@EveryShipout@Output
510                 \fi
511             }%
512         \@PackageInfoNoLine{atbegshi}{Package 'everyshi' patched}%
513         \begingroup
514         \fi
515     \fi
516 \endgroup
517 \let\AtBegShi@PatchEveryshi\relax
518 }
519 \@ifpackageloaded{everyshi}{%
520     \AtBegShi@PatchEveryshi
521 }{%
522     \AtBeginDocument{\AtBegShi@PatchEveryshi}%
523 }

```

### 3.5.3 Class memoir

Fix of method and box.

```

524 \def\AtBegShi@PatchMemoir{%
525     \begingroup
526     \def\AtBegShi@Memoir@shipout{%
527         \afterassignment\mem@shipi
528         \setbox\@cclv=%
529     }%
530     \def\AtBegShi@Memoir@shipi{%
531         \ifvoid\@cclv
532         \expandafter\aftergroup
533         \fi
534         \mem@shipii
535     }%
536     \def\AtBegShi@Memoir@shipiiA{%
537         \mem@oldshipout\vbox{%
538             \trimmarks
539             \unvbox\@cclv
540         }%
541     }%
542     \def\AtBegShi@Memoir@shipiiB{%
543         \ifvoid\@cclv
544         \mem@oldshipout\box\@cclv

```

```

545 \else
546 \mem@oldshipout\vbox{%
547 \trimmarks
548 \unvbox\@cclv
549 }%
550 \fi
551 }%
552 \ifx\AtBegShi@Memoir@shipi\mem@shipi
553 \ifcase\ifx\AtBegShi@Memoir@shipiiA\mem@shipii
554 \z@
555 \else
556 \ifx\AtBegShi@Memoir@shipiiB\mem@shipii
557 \z@
558 \else
559 \@ne
560 \fi
561 \fi
562 \let\AtBegShi@found\relax
563 \ifx\shipout\AtBegShi@Memoir@shipout
564 \def\AtBegShi@found{\shipout}%
565 \else\ifx\AtBegShi@OrgShipout\AtBegShi@Memoir@shipout
566 \def\AtBegShi@found{\AtBegShi@OrgShipout}%
567 \else\ifx\CROP@shipout\AtBegShi@Memoir@shipout
568 \def\AtBegShi@found{\CROP@shipout}%
569 \else\ifx\GPTorg@shipout\AtBegShi@Memoir@shipout
570 \def\AtBegShi@found{\GPTorg@shipout}%
571 \else\ifx\THBorg@shipout\AtBegShi@Memoir@shipout
572 \def\AtBegShi@found{\THBorg@shipout}%
573 \else\ifx\@EveryShipout@Org@Shipout\AtBegShi@Memoir@shipout
574 \def\AtBegShi@found{\@EveryShipout@Org@Shipout}%
575 \fi\fi\fi\fi\fi\fi
576 \ifx\AtBegShi@found\relax
577 \else
578 \expandafter\endgroup
579 \expandafter\def\AtBegShi@found{%
580 \edef\AtBegShi@GroupLevel{\number\currentgrouplevel}%
581 \afterassignment\mem@shipi
582 \setbox\AtBeginShipoutBox=%
583 }%
584 \def\mem@shipi{%
585 \ifnum\AtBegShi@GroupLevel=\currentgrouplevel
586 \else
587 \expandafter\aftergroup
588 \fi
589 \mem@shipii
590 }%
591 \def\mem@shipii{%
592 \ifvoid\AtBeginShipoutBox
593 \else
594 \setbox\AtBeginShipoutBox=\vbox{%
595 \trimmarks
596 \ifvbox\AtBeginShipoutBox
597 \unvbox\AtBeginShipoutBox
598 \else
599 \box\AtBeginShipoutBox
600 \fi
601 }%
602 \expandafter\mem@oldshipout
603 \expandafter\box
604 \expandafter\AtBeginShipoutBox
605 \fi
606 }%

```

```

607         \@PackageInfoNoLine{atbegshi}{Class 'memoir' patched}%
608         \begingroup
609         \fi
610         \fi
611         \fi
612     \endgroup
613     \let\AtBegShi@PatchMemoir\relax
614 }
615 \@ifclassloaded{memoir}{%
616     \AtBegShi@PatchMemoir
617 }{%
618     \AtBeginDocument{\AtBegShi@PatchMemoir}%
619 }
620 \AtBegShi@AtEnd
621 \endpackage

```

## 4 Test

### 4.1 Catcode checks for loading

```

622 (*test1)
623 \catcode'\@=11 %
624 \def\RestoreCatcodes{}
625 \count@=0 %
626 \loop
627     \edef\RestoreCatcodes{%
628         \RestoreCatcodes
629         \catcode\the\count@=\the\catcode\count@\relax
630     }%
631 \ifnum\count@<255 %
632     \advance\count@\@ne
633 \repeat
634
635 \def\RangeCatcodeInvalid#1#2{%
636     \count@=#1\relax
637     \loop
638         \catcode\count@=15 %
639     \ifnum\count@<#2\relax
640         \advance\count@\@ne
641     \repeat
642 }
643 \def\Test{%
644     \RangeCatcodeInvalid{0}{47}%
645     \RangeCatcodeInvalid{58}{64}%
646     \RangeCatcodeInvalid{91}{96}%
647     \RangeCatcodeInvalid{123}{255}%
648     \catcode'\@=12 %
649     \catcode'\=0 %
650     \catcode'\{=1 %
651     \catcode'\}=2 %
652     \catcode'\#=6 %
653     \catcode'\[=12 %
654     \catcode'\]=12 %
655     \catcode'\%=14 %
656     \catcode'\ =10 %
657     \catcode13=5 %
658     \input atbegshi.sty\relax
659     \RestoreCatcodes
660 }
661 \Test
662 \csname @@end\endcsname

```

```

663 \end
664 </test1>
665 <test2>
666 \input atbegshi.sty\relax
667 \def\msg#\{ \immediate\write16}
668 \msg{File: atbegshi-test2.tex 2007/09/09 v1.6 Test file for plain-TeX}
669 \def\testmsg#1#2{%
670   \msg{ }%
671   \msg{*** Test with box (#1), expected page output [#2]]}% hash-ok
672 }
673
674 \newbox\voidbox
675 \def\void{\box\voidbox}
676 \begingroup
677   \setbox\voidbox=\void
678 \endgroup
679
680 \count0=0\relax
681 \AtBeginShipout{%
682   \global\advance\count0 by 1\relax
683   \msg{* Inside \string\AtBeginShipout: [\the\count0]}%
684 }
685
686 \AtBeginShipoutFirst{%
687   \msg{* Inside \string\AtBeginShipoutFirst}%
688   Hello World%
689 }
690
691 \testmsg{\string\null}{1}
692 \shipout\null
693
694 \AtBeginShipoutFirst{%
695   This is too late%
696 }
697
698 \testmsg{void}{ }
699 \shipout\void
700
701 \testmsg{\string\copy255 (not void)}{2}
702 \setbox255\hbox{\vrule height 10bp width 10bp}
703 \shipout\copy255 %
704
705 \testmsg{\string\copy255 (again)}{3}
706 \shipout\copy255 %
707
708 \testmsg{\string\box255}{4}
709 \shipout\box255 %
710
711 \testmsg{\string\box255 (again)}{ }
712 \shipout\box255 %
713
714 \testmsg{\string\hbox}{5}
715 \shipout\hbox{\vrule height 5bp width 20bp}
716
717 \testmsg{\string\vbox}{6}
718 \shipout\vbox{\hrule height 20bp width 5bp}
719
720 \testmsg{\string\null, voided by hook}{ }
721 \def\VoidBox{%
722   \begingroup
723     \setbox\AtBeginShipoutBox=\box\AtBeginShipoutBox
724   \endgroup

```

```

725 }
726 \AtBeginShipout{\VoidBox}
727 \shipout\null
728 \def\VoidBox{}
729
730 \msg{*** \string\beginngroup}
731 \beginngroup
732   \testmsg{void}{}%
733   \shipout\void
734 \msg{*** \string\endgroup}
735 \endgroup
736
737 \msg{*** \string\beginngroup}
738 \beginngroup
739   \testmsg{void}{}%
740   \shipout\void
741   \testmsg{\string\null}{8}%
742   \shipout\null
743 \msg{*** \string\endgroup}
744 \endgroup
745
746 \testmsg{output routine}{9}
747 Hello World
748 \vfill
749 \eject
750
751 \testmsg{\string\null\space(discarded)}{}
752 \AtBeginShipout{%
753   \msg{* Inside \string\AtBeginShipout: DISCARD}%
754   \AtBeginShipoutDiscard
755 }
756 \shipout\null
757
758 \end
759 </test2>
760 <*test3>
761 \NeedsTeXFormat{LaTeX2e}
762 \ProvidesFile{atbegshi-test3.tex}[2007/09/09 v1.6 Test file for LaTeX]
763 \RequirePackage{color}
764 \pagecolor{yellow}
765 \documentclass[a5paper,showtrims]{memoir}
766 \usepackage{atbegshi}
767 \AtBeginShipout{%
768   \setbox\AtBeginShipoutBox=\vbox{%
769     \vbox to 0pt{%
770       \kern-1.5in %
771       \hbox to 0pt{%
772         \kern-1.5in %
773         \color{blue}%
774         \rule{1in}{1in}%
775         \hss
776       }%
777       \vss
778     }%
779     \hrule
780     \hbox{\vrule\box\AtBeginShipoutBox\vrule}%
781     \hrule
782   }%
783 }
784 \usepackage{eso-pic}
785 \makeatletter
786 \@EveryShipout@Init

```

```

787 \let\@EveryShipout@Init\relax
788 \makeatother
789 \AddToShipoutPicture{%
790   \hspace{.52\paperwidth}%
791   \colorbox{cyan}{%
792     \rule{0mm}{\paperheight}%
793     \hspace{.48\paperwidth}%
794   }%
795 }

```

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

```

796 \expandafter\let\csname ver@crop.sty\endcsname\relax
797 \usepackage[color=red,cross,a4,center]{crop}
798 \begin{document}
799 \shipout\null
800 \shipout\box\csname voidb@x\endcsname
801 \section{Hello World}
802 \end{document}
803 </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:macros/latex/contrib/oberdiek/oberdiek-tds.zip](#)

*TDS* refers to the standard “A Directory Structure for T<sub>E</sub>X 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-T<sub>E</sub>X:

```
tex atbegshi.dtx
```

---

<sup>1</sup><http://ftp.ctan.org/tex-archive/>

**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-example.tex → doc/latex/oberdiek/atbegshi-example.tex
atbegshi-test1.tex → doc/latex/oberdiek/atbegshi-test1.tex
atbegshi-test2.tex → doc/latex/oberdiek/atbegshi-test2.tex
atbegshi-test3.tex → doc/latex/oberdiek/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  $\text{\TeX}$  distribution (`te $\text{\TeX}$` , `mik $\text{\TeX}$` , ...) relies on file name databases, you must refresh these. For example, `te $\text{\TeX}$`  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  $\text{\LaTeX}$ .** The `.dtx` chooses its action depending on the format:

**plain- $\text{\TeX}$ :** Run `docstrip` and extract the files.

**$\text{\LaTeX}$ :** Generate the documentation.

If you insist on using  $\text{\LaTeX}$  for `docstrip` (really, `docstrip` does not need  $\text{\LaTeX}$ ), 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 `pdf $\text{\LaTeX}$` :

```
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  $\varepsilon$ -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.

## 7 Index

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

Symbols			
<code>\#</code>	..... 652	<code>\@undefined</code>	..... 123
<code>\%</code>	..... 655	<code>\[</code>	..... 653
<code>\@</code>	..... 623, 648	<code>\]</code>	..... 649
<code>\@EveryShipout@Init</code>	..... 786, 787	<code>\{</code>	..... 650
<code>\@EveryShipout@Org@Shipout</code>	..... 399, 400, 573, 574	<code>\}</code>	..... 651
<code>\@EveryShipout@Output</code>	.. 463, 465, 509	<code>\]</code>	..... 654
<code>\@EveryShipout@Shipout</code>	.... 485, 486	<code>\sqcup</code>	..... 656
<code>\@EveryShipout@Test</code>	458, 468, 495, 498	<b>A</b>	
<code>\@PackageInfoNoLine</code>	184, 441, 512, 607	<code>\AddToShipoutPicture</code>	..... 789
<code>\@PackageWarning</code>	..... 175, 240, 264	<code>\advance</code>	..... 632, 640, 682
<code>\@ccclv</code>	.... 371, 374, 384, 459, 462, 508, 528, 531, 539, 543, 544, 548	<code>\afterassignment</code>	..... 160, 370, 413, 458, 495, 527, 581
<code>\@empty</code>	..... 201, 202, 220	<code>\aftergroup</code>	..... 167, 375, 419, 463, 501, 532, 587
<code>\@ifclassloaded</code>	..... 615	<code>\AtBeginDocument</code>	.. 284, 453, 522, 618
<code>\@ifdefinable</code>	..... 138	<code>\AtBeginShipout</code>	..... 2, 6, 227, 681, 683, 726, 752, 753, 767
<code>\@ifpackageloaded</code>	..... 450, 519		
<code>\@ne</code>	..... 121, 124, 559, 632, 640		

\AtBeginShipoutBox .....	\AtBegShi@Test .....
..... 157, 162, 174, 187, 193,	160, 164
212, 217, 263, 274, 341, 346,	\AtBegShi@vorigin .... 294, 297, 345
353, 414, 428, 431, 433, 438,	
496, 506, 508, 582, 592, 594,	<b>B</b>
596, 597, 599, 604, 723, 768, 780	\baselineskip ..... 213, 382, 426
\AtBeginShipoutDiscard .....	\begin ..... 11, 798
..... 3, 29, 142, 268, 754	\box ..... 187, 193, 353, 384, 428,
\AtBeginShipoutFirst .....	437, 508, 544, 599, 603, 675,
... 3, 221, 235, 242, 686, 687, 694	708, 709, 711, 712, 723, 780, 800
\AtBeginShipoutInit .. 3, 272, 282, 284	<b>C</b>
\AtBeginShipoutNext ... 2, 14, 28, 231	\catcode . 38, 39, 40, 41, 42, 66, 67,
\AtBeginShipoutUpperLeft 3, 7, 15, 340	68, 69, 70, 71, 72, 73, 91, 93, 97,
\AtBegShi@AbortIfUndefined .....	99, 149, 150, 152, 153, 197, 198,
..... 356, 363, 364, 365, 366	623, 629, 638, 648, 649, 650,
\AtBegShi@AddHook .... 229, 233, 256	651, 652, 653, 654, 655, 656, 657
\AtBegShi@AddTo ..... 237, 245, 257	\circle ..... 8
\AtBegShi@AtEnd ..... 95, 96, 359, 620	\color ..... 16, 773
\AtBegShi@BeginPicture 311, 327, 347	\colorbox ..... 791
\AtBegShi@CheckDefinable .....	\copy . 216, 217, 337, 701, 703, 705, 706
. 117, 142, 227, 231, 235, 272, 275	\count ..... 680, 682, 683
\AtBegShi@Crop@ship ..... 386, 393	\count@ ..... 625,
\AtBegShi@Crop@ship ..... 373, 391	629, 631, 632, 636, 638, 639, 640
\AtBegShi@Crop@shiplist ... 379, 392	\CROP@ship ..... 377, 393, 421, 430
\AtBegShi@Crop@shipout .....	\CROP@kernel ..... 383, 427
. 369, 395, 397, 399, 401, 403, 405	\CROP@ship ..... 370, 391, 413, 416
\AtBegShi@Discardfalse ... 179, 185	\CROP@shiplist .... 388, 392, 423, 434
\AtBegShi@Discardtrue ..... 145	\CROP@shipout .....
\AtBegShi@EndPicture .. 321, 332, 349	..... 387, 436, 474, 475, 567, 568
\AtBegShi@Everyshi@Output .. 503, 505	\csname ..... 43,
\AtBegShi@Everyshi@shipout . 457,	53, 74, 87, 90, 112, 118, 148,
470, 472, 474, 476, 478, 480, 485	178, 246, 274, 281, 287, 301,
\AtBegShi@Everyshi@Test ... 461, 468	304, 315, 358, 483, 662, 796, 800
\AtBegShi@First ..... 191, 199	\currentgrouplevel ..... 159,
\AtBegShi@FirstDisabled ... 221, 239	166, 412, 417, 494, 499, 580, 585
\AtBegShi@found ... 394, 396, 398,	<b>D</b>
400, 402, 404, 406, 408, 411,	\deadcycles ..... 144
469, 471, 473, 475, 477, 479,	\documentclass ..... 2, 765
481, 486, 490, 493, 562, 564,	\dp ..... 211, 336
566, 568, 570, 572, 574, 576, 579	
\AtBegShi@GroupLevel ..... 159,	<b>E</b>
166, 412, 417, 494, 499, 580, 585	\E ..... 198
\AtBegShi@Hook ..... 180, 224, 229	\eject ..... 749
\AtBegShi@HookFirst 202, 206, 226, 237	\empty ..... 47
\AtBegShi@HookNext . 181, 182, 225, 233	\end ..... 34, 663, 758, 802
\AtBegShi@horigin .... 293, 296, 343	\endcsname ..... 43,
\AtBegShi@Item ..... 257, 259	53, 74, 87, 90, 112, 118, 148,
\AtBegShi@Memoir@shipi ... 530, 552	178, 246, 274, 281, 287, 301,
\AtBegShi@Memoir@shipiiA ... 536, 553	304, 315, 358, 483, 662, 796, 800
\AtBegShi@Memoir@shipiiB ... 542, 556	\endinput ..... 62, 360
\AtBegShi@Memoir@shipout .....	\endpicture ..... 322
. 526, 563, 565, 567, 569, 571, 573	\errmessage ..... 129
\AtBegShi@OrgProtect .. 177, 189, 192	
\AtBegShi@OrgShipout ... 193, 275,	<b>F</b>
276, 397, 398, 472, 473, 565, 566	\fill ..... 25
\AtBegShi@Output ..... 169, 171	
\AtBegShi@PatchCrop 367, 448, 451, 453	<b>G</b>
\AtBegShi@PatchEveryshi .....	\g@addto@macro ..... 254
..... 455, 517, 520, 522	\gdef ..... 182, 224, 225, 226, 278
\AtBegShi@PatchMemoir .....	\GPTorg@shipout .....
..... 524, 613, 616, 618	..... 401, 402, 476, 477, 569, 570
\AtBegShi@Shipout ..... 155, 277	

<b>H</b>	
\hbox	157, 328, 341, 702, 714, 715, 771, 780
\hrule	718, 779, 781
\hspace	790, 793
\hss	775
\ht	210, 335, 346
<b>I</b>	
\ifAtBegShi@Discarded	141, 183, 260
\ifcase	44, 120, 300, 553
\ifdim	165
\ifnum	166, 417, 499, 585, 631, 639
\ifpdf	292
\ifvbox	596
\ifvoid	174, 263, 374, 431, 462, 506, 531, 543, 592
\ifx	45, 47, 53, 74, 82, 112, 118, 120, 123, 148, 202, 246, 281, 287, 301, 304, 315, 358, 391, 392, 393, 395, 397, 399, 401, 403, 405, 408, 468, 470, 472, 474, 476, 478, 480, 483, 485, 490, 552, 553, 556, 563, 565, 567, 569, 571, 573, 576
\ignorespaces	319, 330
\immediate	55, 76, 667
\input	113, 288, 658, 666
<b>K</b>	
\kern	158, 343, 345, 346, 770, 772
<b>L</b>	
\lastkern	165
\line	17, 18
\lineskip	214, 380, 424
\lineskiplimit	215, 381, 425
\loop	626, 637
<b>M</b>	
\makeatletter	785
\makeatother	788
\mem@oldshipout	405, 406, 480, 481, 537, 544, 546, 602
\mem@shipi	527, 552, 581, 584
\mem@shipii	534, 553, 556, 589, 591
\MessageBreak	241, 265
\msg	667, 668, 670, 671, 683, 687, 730, 734, 737, 743, 753
<b>N</b>	
\NeedsTeXFormat	761
\newbox	674
\newif	141
\newpage	13, 22, 27, 32
\null	691, 692, 720, 727, 741, 742, 751, 756, 799
\number	159, 412, 494, 580
<b>P</b>	
\p@	158
\PackageInfo	58
\pagecolor	764
\paperheight	8, 17, 18, 792
\paperwidth	8, 17, 18, 790, 793
\par	24
\pdfhorigin	293
\pdfvorigin	294
\picture	313
\protect	177, 189, 192
\ProvidesFile	762
\ProvidesPackage	88
\put	8, 17, 18
<b>R</b>	
\RangeCatcodeInvalid	635, 644, 645, 646, 647
\repeat	633, 641
\RequirePackage	115, 290, 763
\RestoreCatcodes	624, 627, 628, 659
\rlap	342
\rule	774, 792
<b>S</b>	
\section	12, 801
\setbox	157, 162, 187, 204, 212, 328, 341, 371, 414, 433, 459, 496, 508, 528, 582, 594, 677, 702, 723, 768
\shipout	276, 277, 395, 396, 470, 471, 563, 564, 692, 699, 703, 706, 709, 712, 715, 718, 727, 733, 740, 742, 756, 799, 800
\space	130, 751
<b>T</b>	
\Test	643, 661
\testmsg	669, 691, 698, 701, 705, 708, 711, 714, 717, 720, 732, 739, 741, 746, 751
\THBorg@shipout	403, 404, 478, 479, 571, 572
\the	91, 97, 250, 629, 683
\TMP@EnsureCode	94, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110
\toks	249, 250
\trimmarks	538, 547, 595
<b>U</b>	
\unitlength	317
\unvbox	539, 548, 597
\usepackage	3, 4, 5, 766, 784, 797
<b>V</b>	
\vbox	204, 212, 344, 387, 433, 537, 546, 594, 717, 718, 768, 769
\vfill	748
\void	675, 677, 699, 733, 740
\VoidBox	721, 726, 728
\voidbox	674, 675, 677
\vrule	702, 715, 780
\vspace	25
\vss	350, 777
<b>W</b>	
\wd	209

\write .....	55, 76, 667		<b>Z</b>
		\z@ ...	126, 144, 165, 204, 209, 210,
	<b>X</b>		211, 215, 216, 249, 250, 380,
\X .....	197		381, 382, 424, 425, 426, 554, 557
\x ....	43, 45, 47, 54, 58, 60, 75, 80, 87	\z@skip .....	213, 214