

The `ltshipout` package*

Frank Mittelbach

September 11, 2020

Contents

1	Introduction	1
1.1	Overloading the <code>\shipout</code> primitive	2
1.2	Provided hooks	3
1.3	Special commands for use inside the hooks	4
1.4	Information counters	5
1.5	Debugging shipout code	5
2	Emulating commands from other packages	5
2.1	Emulating <code>atbegshi</code>	6
2.2	Emulating <code>everyshi</code>	6
2.3	Emulating <code>atenddvi</code>	7
2.4	Emulating <code>everypage</code>	7
3	The Implementation	7
3.1	Debugging	7
3.2	Handling the end of job hook	16
4	Legacy \LaTeX 2ϵ interfaces	19
5	Internal commands needed elsewhere	19
6	Package emulation for compatibility	21
6.1	Package <code>atenddvi</code> emulation	21
6.2	Package <code>atbegshi</code> emulation	21
6.3	Package <code>everyshi</code> emulation	23
	Index	24

*This package has version v1.0b dated 2020/09/09, © \LaTeX Project.

1 Introduction

The code provides an interface to the `\shipout` primitive of T_EX which is called when a finished pages is finally “shipped out” to the target output file, e.g., the `.dvi` or `.pdf` file. A good portion of the code is based on ideas by Heiko Oberdiek implemented in his packages `atbegshi` and `atenddvi` even though the interfaces are somewhat different.¹

1.1 Overloading the `\shipout` primitive

`\shipout`

With this implementation T_EX’s `shipout` primitive is no longer available for direct use. Instead `\shipout` is running some (complicated) code that picks up the box to be shipped out regardless of how that is done, i.e., as a constructed `\vbox` or `\hbox` or as a box register.

It then stores it in a named box register. This box can then be manipulated through a set of hooks after which it is shipped out for real.

`\ShipoutBox`
`\l_shipout_box`

This box register is called `\ShipoutBox` (alternatively available via the L3 name `\l_shipout_box`).

`\l_shipout_box_ht_dim`
`\l_shipout_box_dp_dim`
`\l_shipout_box_wd_dim`
`\l_shipout_box_ht_plus_dp_dim`

The `shipout` box dimensions are available in the L3 registers `\l_shipout_box_ht_dim`, etc. (there are no L^AT_EX 2_ε names).² These variables can be used inside the hook code for `shipout/before`, `shipout/foreground` and `shipout/background` if needed.

¹Heiko’s interfaces are emulated by the kernel code, if a document requests his packages, so older documents will continue to work.

²Might need changing, but HO’s version as strings is not really helpful I think).

1.2 Provided hooks

`shipout/before`
`shipout/foreground`
`shipout/background`
`shipout/firstpage`
`shipout/lastpage`

The code offers a number of hooks into which packages (or the user) can add code to support different use cases. These are:

shipout/before This hook is executed after the finished page has been stored in `\ShipoutBox` / `\l_shipout_box`. It can be used to alter that box content or to discard it completely (see `\DiscardShipoutBox` below).

shipout/background This hook adds a picture environment into the background of the page with the (0,0) coordinate in the top-left corner using a `\unitlength` of `1pt`. It should therefore only receive `\put` commands or other commands suitable in a `picture` environment and the vertical coordinate values would normally be negative.

Technically this is implemented by adding a zero-sized `\hbox` as the very first item into the `\ShipoutBox` containing that `picture` environment. Thus the rest of the box content will overprint what ever is typeset by that hook.

shipout/foreground This hook adds a picture environment into the foreground of the page with the (0,0) coordinate in the top-left corner using a `\unitlength` of `1pt`.

Technically this is implemented by adding a zero-sized `\hbox` as the very last item into the `\ShipoutBox` and raising it up so that it still has its (0,0) point in the top-left corner. But being placed after the main box content it will be typeset later and thus overprints it (i.e., is in the foreground).

shipout/firstpage The material from this hook is executed only once at the very beginning of the first output page that is shipped out (i.e., not discarded at the last minute). It should only contain `\special` or similar commands needed to direct post processors handling the `.dvi` or `.pdf` output.³

shipout/lastpage The corresponding hook to add `\specials` at the very end of the output file. It is only executed on the very last page — or rather on the page that `LATEX` believes is the last one.

It may not be possible for `LATEX` to correctly determine which page is the last one without several reruns. If this happens and the hook is non-empty then `LATEX` will add an extra page to place the material and also request a rerun to get the correct placement sorted out.

As mentioned above the hook `shipout/before` is executed first and can manipulate the prepared shipout box stored in `\ShipoutBox` or set things up for use in `\write` during the actual shipout. The other hooks are added inside `hboxes` to the box being shipped out in the following order:

³In `LATEX 2ε` that was already existing, but implemented using a box register with the name `\@begindvibox`.

<code>shipout/firstpage</code>	only on the first page
<code>shipout/background</code>	
<code><boxed content of \ShipoutBox></code>	
<code>shipout/foreground</code>	
<code>shipout/lastpage</code>	only on the last page

If any of the hooks has no code then that particular no box is added at that point.

In a document that doesn't produce pages, e.g., only makes `\typeouts`, none of the hooks are executed (as there is no `\shipout`) not even the `shipout/lastpage` hook.

`\AtBeginDvi`
`\AtEndDvi`

`\AtBeginDvi` is the existing L^AT_EX 2_ε interface to fill the `shipout/firstpage` hook. This is not really a good name as it is not just supporting `.dvi` but also `.pdf` output or `.dvi`.

`\AtEndDvi` is the counterpart that was not available in the kernel but only through the package `atenddvi`. It fills the `shipout/lastpage` hook.

Todo: better names? Any suggestions?

1.3 Special commands for use inside the hooks

`\DiscardShipoutBox`
`\shipout_discard_box:`

`\AddToHookNext {shipout/before} {...\DiscardShipoutBox...}`

The `\DiscardShipoutBox` declaration (L3 name `\shipout_discard_box:`) requests that on the next shipout the page box is thrown away instead of being shipped to the `.dvi` or `.pdf` file.

Typical applications wouldn't do this unconditionally, but have some processing logic that decides to use or not to use the page.

Note that if this declaration is used directly in the document it may depend on the placement to which page it applies, given that L^AT_EX output routine is called in an asynchronous manner!

Todo: Once we have a new mark mechanism available we can improve on that and make sure that the declaration applies to the page that contains it.

In the `atbegshi` package there are a number of additional commands for use inside the `shipout/before` hook. They should normally not be needed any more as one can instead simply add code to the hooks `shipout/before`, `shipout/background` or `shipout/foreground`.⁴ If `atbegshi` gets loaded then those commands become available as public functions with their original names as given below.

⁴If that assumption turns out to be wrong it would be trivial to change them to public functions (right now they are private).

1.4 Information counters

<code>\ReadonlyShipoutCounter</code>	<code>\ifnum\ReadOnlyShipoutCounter=...</code>
<code>\g_shipout_readonly_int</code>	<code>\int_use:N \g_shipout_readonly_int % expl3 usage</code>

This integer holds the number of pages shipped out up to now (including the one to be shipped out when inside the output routine). More precisely, it is incremented only after it is clear that a page will be shipped out, i.e., after the `shipout/before` hook (because that might discard the page)!

Just like with the `page` counter its value is only accurate within the output routine. In the body of the document it may be off by one as the output routine is called asynchronously!

Also important: it *must not* be set, only read. There are no provisions to prevent that but if you do, chaos will be the result. To emphasize this fact it is not provided as a \LaTeX counter but as a \TeX counter (i.e., a command), so `\Alph{\ReadonlyShipoutCounter}` etc, would not work.

<code>totalpages</code>	<code>\arabic{totalpages}</code>
<code>\g_shipout_totalpages_int</code>	<code>\int_use:N \g_shipout_totalpage_int % expl3 usage</code>

In contrast to `\ReadonlyShipoutCounter`, the `totalpages` counter is a \LaTeX counter and incremented for each shipout attempt including those pages that are discarded for one or the other reason. Again `shipout/before` sees the counter before it is incremented).

Furthermore, while it is incremented for each page, its value is never used by \LaTeX . It can therefore be freely reset or changed by user code, for example, to additionally count a number of pages that are not build by \LaTeX but are added in a later part of the process, e.g., cover pages or picture pages made externally.

Important: as this is a page-related counter its value is only reliable inside the output routine!

<code>\PreviousTotalPages</code>	<code>\thetotalpages/\PreviousTotalPages</code>
----------------------------------	---

Command that expands to the number of total pages from the previous run. If there was no previous run or if used in the preamble it expands to 0. Note that this is a command to a counter, so in order to display the number in, say, Roman numerals you have to assign its value to a counter and then use `\Roman` on that counter.

1.5 Debugging shipout code

<code>\DebugShipoutsOn</code>	<code>\DebugShipoutsOn</code>
<code>\DebugShipoutsOff</code>	Turn the debugging of shipout code on or off. This displays changes made to the shipout data structures.
<code>\shipout_debug_on:</code>	
<code>\shipout_debug_off:</code>	

Todo: This needs some rationalizing and may not stay this way.

2 Emulating commands from other packages

The packages in this section are no longer necessary but as they are used in other packages they are emulated when they are loaded via `\usepackage` or `\RequirePackage`.

2.1 Emulating atbegshi

<code>\AtBeginShipoutUpperLeft</code>	<code>\AddToHook {shipout/before}</code>
<code>\AtBeginShipoutUpperLeftForeground</code>	<code>{... \AtBeginShipoutUpperLeft{<code>}...}</code>

This adds a `picture` environment into the background of the shipout box expecting `<code>` to contain `picture` commands. The same effect can be obtained by simply using kernel features as follows:

```
\AddToHook{shipout/background}{<code>}
```

There is one technical difference: if `\AtBeginShipoutUpperLeft` is used several times each invocation is put into its own box inside the shipout box whereas all `<code>` going into `shipout/background` ends up all in the same box in the order it is added or sorted based on the rules for the hook chunks.

`\AtBeginShipoutUpperLeftForeground` is similar with the difference that the `picture` environment is placed in the foreground. To model it with the kernel functions use the hook `shipout/foreground` instead.

<code>\AtBeginShipoutAddToBox</code>	<code>\AddToHook {shipout/before} {... \AtBeginShipoutAddToBox{<code>}...}</code>
<code>\AtBeginShipoutAddToBoxForeground</code>	

These work like `\AtBeginShipoutUpperLeft` and `\AtBeginShipoutUpperLeftForeground` with the difference that `<code>` is directly placed into an `\hbox` inside the shipout box and not surrounded by a `picture` environment.

To emulate them using `shipout/background` or `shipout/foreground` you may have to wrap `<code>` into a `\put` statement but if the code is not doing any typesetting just adding it to the hook should be sufficient.

<code>\AtBeginShipoutBox</code>	This is the name of the shipout box as <code>atbegshi</code> knows it.
---------------------------------	--

<code>\AtBeginShipoutInit</code>	By default <code>atbegshi</code> delayed its action until <code>\begin{document}</code> . This command was forcing it in an earlier place. With the new concept it does nothing.
----------------------------------	--

<code>\AtBeginShipout</code>	<code>\AtBeginShipout{<code>} ≡ \AddToHook{shipout/before}{<code>}</code>
<code>\AtBeginShipoutNext</code>	<code>\AtBeginShipoutNext{<code>} ≡ \AddToHookNext{shipout/before}{<code>}</code>

This is equivalent to filling the `shipout/before` hook by either using `\AddToHook` or `\AddToHookNext`, respectively.

<code>\AtBeginShipoutFirst</code>	The <code>atbegshi</code> names for <code>\AtBeginDvi</code> and <code>\DiscardShipoutBox</code> .
<code>\AtBeginShipoutDiscard</code>	

2.2 Emulating everyshi

<code>\EveryShipout</code>	<code>\EveryShipout{<code>} ≡ \AddToHook{shipout/before}{<code>}</code>
----------------------------	---

<code>\AtNextShipout</code>	<code>\AtNextShipout{<code>} ≡ \AddToHookNext{shipout/before}{<code>}</code>
-----------------------------	--

2.3 Emulating atenddvi

The atenddvi package implemented only a single command: \AtEndDvi and that is now available out of the box.

2.4 Emulating everypage

This page takes over the original \@begindvi hook and replaces it. It should be all covered by the hooks offered here (details need checking) and thus could simply use the provided hooks rather than defining its own.

3 The Implementation

```
1 <@=shipout>
```

At the moment the whole module rolls back in one go, but if we make any modifications in later releases this will then need splitting.

```
2 <*2ekernel | latexrelease>
3 <latexrelease> \IncludeInRelease{2020/10/01}%
4 <latexrelease>           {\shipout}{Hook mangement (shipout)}%
5 \ExplSyntaxOn
```

3.1 Debugging

```
\g__shipout_debug_bool Holds the current debugging state.
6 \bool_new:N \g__shipout_debug_bool
(End definition for \g__shipout_debug_bool.)
```

```
\shipout_debug_on: Turns debugging on and off by redefining \__shipout_debug:n.
\shipout_debug_off:
  \__shipout_debug:n
\__shipout_debug_gset:
7 \cs_new_eq:NN \__shipout_debug:n \use_none:n
8 \cs_new_protected:Npn \shipout_debug_on:
9   {
10     \bool_gset_true:N \g__shipout_debug_bool
11     \__shipout_debug_gset:
12   }
13 \cs_new_protected:Npn \shipout_debug_off:
14   {
15     \bool_gset_false:N \g__shipout_debug_bool
16     \__shipout_debug_gset:
17   }
18 \cs_new_protected:Npn \__shipout_debug_gset:
19   {
20     \cs_gset_protected:Npx \__shipout_debug:n ##1
21     { \bool_if:NT \g__shipout_debug_bool {##1} }
22   }
```

(End definition for \shipout_debug_on: and others. These functions are documented on page 5.)

```
\ShipoutBox The box filled with the page to be shipped out (both L3 and LATEX 2ε name).
\l_shipout_box
23 \box_new:N \l_shipout_box
24 \cs_set_eq:NN \ShipoutBox \l_shipout_box
```

(End definition for \ShipoutBox and \l_shipout_box. These functions are documented on page 2.)

`__shipout_execute:` This is going to be the code run by `\shipout`. The code follows closely the ideas from `atbegshi`, so not documenting that here for now.

```

25 \cs_set:Npn\__shipout_execute: {
26   \tl_set:Nx \l__shipout_group_level_tl
27     { \int_value:w \tex_currentgrouplevel:D }
28   \tex_afterassignment:D \__shipout_execute_test_level:
29   \tex_setbox:D \l_shipout_box
30 }

```

(End definition for `__shipout_execute:.`)

`\shipout` Overloading the `\shipout` primitive:

```

31 \cs_gset_eq:NN \shipout \__shipout_execute:

```

(End definition for `\shipout`. This function is documented on page 2.)

`\l__shipout_group_level_tl` Helper token list to record the group level at which `__shipout_execute:` is encountered.

```

32 \tl_new:N \l__shipout_group_level_tl

```

(End definition for `\l__shipout_group_level_tl`.)

`_shipout_execute_test_level:` If the group level has changed then we are still constructing `\l_shipout_box` and to continue we need to wait until the current group has finished, hence the `\tex_aftergroup:D`.

```

33 \cs_new:Npn \__shipout_execute_test_level: {
34   \int_compare:nNnT
35     \l__shipout_group_level_tl < \tex_currentgrouplevel:D
36     \tex_aftergroup:D
37   \__shipout_execute_cont:
38 }

```

(End definition for `_shipout_execute_test_level:.`)

`__shipout_execute_cont:` When we have reached this point the shipout box has been processed and is available in `\l_shipout_box` and ready for real ship out (perhaps)..

First we quickly check if it is void (can't happen in the standard L^AT_EX output routine but `\shipout` might be called from a package that has some special processing logic). If it is void we aren't shipping anything out and processing ends.⁵

```

39 \cs_new:Npn \__shipout_execute_cont: {
40   \box_if_empty:NTF \l_shipout_box
41     { \PackageWarning{!tshipout}{Ignoring~ void~ shipout~ box} }
42   {

```

Otherwise we assume that we will ship something and prepare for final adjustments (in particular setting the state of `\protect` while we are running the hook code). We also save the current `\protect` state to restore it later.

```

43     \bool_gset_false:N \g__shipout_discard_bool
44     \cs_set_eq:NN \__shipout_saved_protect: \protect
45     \set@typeset@protect

```

We also store the current shipout box dimension in registers, so that they can be used in the hook code.⁶

```

46     \__shipout_get_box_size:N \l_shipout_box

```

⁵In that case we don't reset the deadcycles, that would be up to the OR processing logic to do.

⁶This is not really necessary as the code could access them via `\box_ht:N`, etc., but it is perhaps convenient.

Then we execute the `shipout/before` hook.

```
47 \hook_use:n {shipout/before}
```

In `\g_shipout_totalpages_int` we count all shipout attempts so we increment that counter already here (the other one is incremented later when we know for sure that we do a `\shipout`).

We increment it after running the above hook so that the values for `\g_shipout_totalpages_int` and `\` are in sync while the hook is executed (in the case that `totalpages` isn't manually altered or through discarding pages that is).

```
48 \int_gincr:N \g_shipout_totalpages_int
```

The above hook might contain code that requests the page to be discarded so we now test for it.

```
49 \bool_if:NTF \g__shipout_discard_bool
50 { \PackageInfo{!tshipout}{Completed~ page~ discarded}
51 \bool_gset_false:N \g__shipout_discard_bool
```

As we are discarding the page box and not shipping anything out, we need to do some house cleaning and reset T_EX's deadcycles so that it doesn't complain about too many calls to the OR without any shipout.

```
52 \tex_deadcycles:D \c_zero_int
```

Todo: In atbegshi the box was dropped but is that actually needed? Or the resetting of `\protect` to its kernel value?

```
53 % \group_begin:
54 % \box_set_eq_drop:NN \l_shipout_box \l_shipout_box
55 % \group_end:
56 % \cs_set_eq:NN \protect \exp_not:N
57 }
```

Even if there was no explicit request to discard the box it is possible that the code for the hook `shipout/before` has voided the box (by mistake or deliberately). We therefore test once more but this time make it a warning, because the best practice way is to use the request mechanism.

```
58 { \box_if_empty:NTF \l_shipout_box
59 { \PackageWarning{!tshipout}{
60 Shipout~ box~ was~ voided~ by~ hook,\MessageBreak
61 ignoring~ shipout~ box }
62 }
```

Finally, if the box is still non-empty we are nearly ready to ship it out. First we increment the total page counter so that we can later test if we have reached the final page according to our available information.⁷

```
63 {
64 \int_gincr:N \g_shipout_readonly_int
65 \__shipout_debug:n {
66 \typeout{Absolute~ page~ =~ \int_use:N \g_shipout_readonly_int
67 \space (target:~ \@abspage@last)}
68 }
```

⁷Doing that earlier would be wrong because we might end up with the last page counted but discard and then we have no place to add the final objects into the output file.

Then we store the box sizes again (as they may have changed) and then look at the hooks `shipout/foreground` and `shipout/background`. If either or both are non-empty we add a `picture` environment to the box (in the foreground and or in the background) and execute the hook code inside that environment.

```

69         \_shipout_get_box_size:N \l_shipout_box
70         \hook_if_empty:nF {shipout/foreground}
71         { \_shipout_add_foreground_picture:n
72           { \hook_use:n {shipout/foreground} } }
73         \hook_if_empty:nF {shipout/background}
74         { \_shipout_add_background_picture:n
75           { \@kernel@before@shipout@background
76             \hook_use:n {shipout/background} } }

```

We then run `_shipout_execute_firstpage_hook`: that adds the content of the hook `shipout/firstpage` to the start of the first page (if non-empty). It is then redefined to do nothing on later pages.

```

77         \_shipout_execute_firstpage_hook:

```

The we check if we have to add the `shipout/lastpage` hook because we have reached the last page. This test will be false for all but one (and hopefully the correct) page.

```

78         \int_compare:nNnT \@abspage@last = \g_shipout_readonly_int
79         { \hook_if_empty:nF {shipout/lastpage}
80           { \_shipout_debug:n { \typeout{Executing~ lastpage~ hook~
81             on~ page~ \int_use:N \g_shipout_readonly_int } }
82             \_shipout_add_foreground_box:n { \UseHook{shipout/lastpage}
83               \@kernel@after@shipout@lastpage }
84           }
85           \bool_gset_true:N \g__shipout_lastpage_handled_bool
86         }

```

Finally we run the actual \TeX primitive for `shipout`. As that will expand delayed `\write` statements inside the page in which protected commands should not expand we first change `\protect` to the appropriate definition for that case.

```

87         \cs_set_eq:NN \protect \exp_not:N
88         \tex_shipout:D \box_use:N \l_shipout_box
89     }
90 }

```

Restore the value of `\protect` in case `\shipout` is called outside of the output routine (where it is automatically restored because of the implicit group).

```

91     \cs_set_eq:NN \protect \_shipout_saved_protect:
92 }
93 }

```

(End definition for `_shipout_execute_cont:`.)

`_shipout_saved_protect:` Remeber the current `\protect` state.

```

94 \cs_new_eq:NN \_shipout_saved_protect: \protect

```

(End definition for `_shipout_saved_protect:`.)

`shipout/before` Declaring all hooks for the `shipout` code.
`shipout/foreground`
`shipout/background`
`shipout/firstpage`
`shipout/lastpage`

```

95 \hook_new:n{shipout/before}
96 \hook_new:n{shipout/foreground}
97 \hook_new:n{shipout/background}

```

```

98 \hook_new:n{shipout/firstpage}
99 \hook_new:n{shipout/lastpage}

```

(End definition for `shipout/before` and others. These functions are documented on page 3.)

```

\kernel@after@shipout@lastpage And here are the internal kernel hooks going before or after the public ones where needed.
\kernel@before@shipout@background 100 \let\kernel@after@shipout@lastpage\empty
101 \let\kernel@before@shipout@background\empty

```

(End definition for `\kernel@after@shipout@lastpage` and `\kernel@before@shipout@background`. These functions are documented on page ??.)

```

\__shipout_execute_firstpage_hook: This command adds any specials into a box and adds that to the very beginning of the
first box shipped out. After that we redefine it to do nothing on later pages.

```

```

102 \cs_new:Npn \__shipout_execute_firstpage_hook: {

```

Adding something to the beginning means adding it to the background as that layer is done first in the output. Of course that is only needed if the hook actually contains anything.

```

103 \hook_if_empty:nF {shipout/firstpage}
104 { \__shipout_add_background_box:n { \UseHook{shipout/firstpage} } }

```

Once we are here we change the definition to do nothing next time and we also change the command used to implement `\AtBeginDvi` to become a warning and not add further material to a hook that is never used again.

```

105 \cs_gset_eq:NN \__shipout_execute_firstpage_hook: \prg_do_nothing:
106 \cs_gset:Npn \__shipout_add_firstpage_material:Nn ##1 ##2 {
107 \latex@warning{
108 First~ page~ is~ already~ shipped~ out,~ ignoring\MessageBreak
109 \string##1 }
110 }
111 }

```

(End definition for `__shipout_execute_firstpage_hook:`.)

```

\g__shipout_lastpage_handled_bool A boolean to signal if we have already handled the shipout/lastpage hook.

```

```

112 \bool_new:N \g__shipout_lastpage_handled_bool

```

(End definition for `\g__shipout_lastpage_handled_bool`.)

```

\__shipout_add_firstpage_material:Nn This command adds material to the shipout/firstpage hook. It is used in \AtBeginDvi,
etc. The first argument is the command through which it is called. Initially this is ignored
but once we are passed the first page it can be used to generate a warning message
mentioning the right user command.

```

```

113 \cs_new:Npn \__shipout_add_firstpage_material:Nn #1#2 {
114 \AddToHook{shipout/firstpage}{#2}
115 }

```

(End definition for `__shipout_add_firstpage_material:Nn`.)

```

\__shipout_get_box_size:N Store the box dimensions in dimen registers.

```

Todo: This could/should perhaps be generalized to set height depth and width given an arbitrary box.

```

116 \cs_new:Npn \__shipout_get_box_size:N #1 {
117   \dim_set:Nn \l_shipout_box_ht_dim { \box_ht:N #1 }
118   \dim_set:Nn \l_shipout_box_dp_dim { \box_dp:N #1 }
119   \dim_set:Nn \l_shipout_box_wd_dim { \box_wd:N #1 }
120   \dim_set:Nn \l_shipout_box_ht_plus_dp_dim { \l_shipout_box_ht_dim +
121                                             \l_shipout_box_dp_dim }
122 }

```

(End definition for `__shipout_get_box_size:N`.)

`\l_shipout_box_ht_dim` And here are the variables set by `__shipout_get_box_size:N`.

```

\l_shipout_box_dp_dim 123 \dim_new:N \l_shipout_box_ht_dim
\l_shipout_box_wd_dim 124 \dim_new:N \l_shipout_box_dp_dim
\l_shipout_box_ht_plus_dp_dim 125 \dim_new:N \l_shipout_box_wd_dim
126 \dim_new:N \l_shipout_box_ht_plus_dp_dim

```

(End definition for `\l_shipout_box_ht_dim` and others. These functions are documented on page 2.)

`\g__shipout_discard_bool` Indicate whether or not the current page box should be discarded

```

127 \bool_new:N \g__shipout_discard_bool

```

(End definition for `\g__shipout_discard_bool`.)

`\l__shipout_tmp_box` We need a box for the background and foreground material and a token register to
`\l__shipout_saved_badness_tl` remember badness settings as we disable them during the buildup below.

```

128 \box_new:N \l__shipout_tmp_box
129 \tl_new:N \l__shipout_saved_badness_tl

```

(End definition for `\l__shipout_tmp_box` and `\l__shipout_saved_badness_tl`.)

`__shipout_add_background_box:n` In standard L^AT_EX the shipout box is always a `\vbox` but here we allow for other usage as well, in case some package has its own output routine.

```

130 \cs_new:Npn \__shipout_add_background_box:n #1
131 { \__shipout_get_box_size:N \l_shipout_box

```

But we start testing for a vertical box as that should be the normal case.

```

132   \box_if_vertical:NTF \l_shipout_box
133   {

```

Save current values of `\vfuzz` and `\vbadness` then change them to allow box manipulations without warnings.

```

134     \tl_set:Nx \l__shipout_saved_badness_tl
135     { \vfuzz=\the\vfuzz\relax
136       \vbadness=\the\vbadness\relax }
137     \vfuzz=\c_max_dim
138     \vbadness=\c_max_int

```

Then we reconstruct `\l_shipout_box` ...

```

139     \vbox_set_to_ht:Nnn \l_shipout_box \l_shipout_box_ht_plus_dp_dim
140     {

```

... the material in #1 is placed into a horizontal box with zero dimensions.

```

141       \hbox_set:Nn \l__shipout_tmp_box
142       { \l__shipout_saved_badness_tl #1 }
143       \box_set_wd:Nn \l__shipout_tmp_box \c_zero_dim
144       \box_set_ht:Nn \l__shipout_tmp_box \c_zero_dim
145       \box_set_dp:Nn \l__shipout_tmp_box \c_zero_dim

```

The we typeset that box followed by whatever was in `\l_shipout_box` before (unpacked).

```

146         \skip_zero:N \baselineskip
147         \skip_zero:N \lineskip
148         \skip_zero:N \lineskiplimit
149         \box_use:N \l__shipout_tmp_box
150         \vbox_unpack:N \l_shipout_box

```

The `\kern` ensures that the box has no depth which is afterwards explicitly corrected.

```

151         \kern \c_zero_dim
152     }
153     \box_set_ht:Nn \l_shipout_box \l_shipout_box_ht_dim
154     \box_set_dp:Nn \l_shipout_box \l_shipout_box_dp_dim

```

Todo: The whole boxing maneuver looks a bit like overkill to me, but for the moment I leave.

```

155     \l__shipout_saved_badness_tl
156 }
157 {

```

A horizontal box is handled in a similar way. The last case would be a void box in which case we do nothing hence the missing F branch.

```

158     \box_if_horizontal:NT \l_shipout_box
159     {
160         \tl_set:Nx \l__shipout_saved_badness_tl
161         { \hfuzz=\the\hfuzz\relax
162           \hbadness=\the\hbadness\relax }
163         \hfuzz=\c_max_dim
164         \hbadness=\c_max_int
165         \hbox_set_to_wd:Nnn \l_shipout_box \l_shipout_box_wd_dim
166         {
167             \hbox_set:Nn \l__shipout_tmp_box
168             { \l__shipout_saved_badness_tl #1 }
169             \box_set_wd:Nn \l__shipout_tmp_box \c_zero_dim
170             \box_set_ht:Nn \l__shipout_tmp_box \c_zero_dim
171             \box_set_dp:Nn \l__shipout_tmp_box \c_zero_dim
172             \box_move_up:nn
173             \l_shipout_box_ht_dim
174             { \box_use:N \l__shipout_tmp_box }
175             \hbox_unpack:N \l_shipout_box
176         }
177         \l__shipout_saved_badness_tl
178     }
179 }
180 }

```

(End definition for `__shipout_add_background_box:n`)

`__shipout_add_foreground_box:n` Foreground boxes are done in the same way, only the order and placement of boxes has to be done differently.

```

181 \cs_new:Npn \__shipout_add_foreground_box:n #1
182 {
183     \box_if_vertical:NTF \l_shipout_box
184     {
185         \tl_set:Nx \l__shipout_saved_badness_tl

```

```

186     { \vfuzz=\the\vfuzz\relax
187       \vbadness=\the\vbadness\relax }
188 \vfuzz=\c_max_dim
189 \vbadness=\c_max_int
190 \vbox_set_to_ht:Nnn \l_shipout_box \l_shipout_box_ht_plus_dp_dim
191   {
192     \hbox_set:Nn \l__shipout_tmp_box
193       { \l__shipout_saved_badness_tl #1 }
194     \box_set_wd:Nn \l__shipout_tmp_box \c_zero_dim
195     \box_set_ht:Nn \l__shipout_tmp_box \c_zero_dim
196     \box_set_dp:Nn \l__shipout_tmp_box \c_zero_dim
197     \skip_zero:N \baselineskip
198     \skip_zero:N \lineskip
199     \skip_zero:N \lineskiplimit
200     \vbox_unpack:N \l_shipout_box
201     \kern -\l_shipout_box_ht_plus_dp_dim
202     \box_use:N \l__shipout_tmp_box
203     \kern \l_shipout_box_ht_plus_dp_dim
204   }
205 \l__shipout_saved_badness_tl
206 \box_set_ht:Nn \l_shipout_box \l_shipout_box_ht_dim
207 \box_set_dp:Nn \l_shipout_box \l_shipout_box_dp_dim
208 }
209 {
210 \box_if_horizontal:NT \l_shipout_box
211   {
212     \tl_set:Nx \l__shipout_saved_badness_tl
213       { \hfuzz=\the\hfuzz\relax
214         \hbadness=\the\hbadness\relax }
215     \hfuzz=\c_max_dim
216     \hbadness=\c_max_int
217     \hbox_set_to_wd:Nnn \l_shipout_box \l_shipout_box_wd_dim
218     {
219       \hbox_unpack:N \l_shipout_box
220       \kern -\box_wd:N \l_shipout_box
221       \hbox_set:Nn \l__shipout_tmp_box
222         { \l__shipout_saved_badness_tl #1 }
223       \box_set_wd:Nn \l__shipout_tmp_box \c_zero_dim
224       \box_set_ht:Nn \l__shipout_tmp_box \c_zero_dim
225       \box_set_dp:Nn \l__shipout_tmp_box \c_zero_dim
226       \box_move_up:nn { \box_ht:N \l_shipout_box }
227         { \box_use:N \l__shipout_tmp_box }
228       \kern \box_wd:N \l_shipout_box
229     }%
230     \l__shipout_saved_badness_tl
231   }
232 }
233 }

```

(End definition for `_shipout_add_foreground_box:n`.)

`_shipout_init_page_origins:` Two constants holding the offset of the top-left with respect to the media box.
`\c__shipout_horigin_tl` Setting the constants this way is courtesy of Bruno.
`\c__shipout_vorigin_tl`

We delay setting the constants to the last possible place as there might be updates in the preamble or even in the `begindocument` hook that affects their setup.

```

234 \cs_new:Npn \__shipout_init_page_origins: {
235   \tl_const:Nx \c__shipout_horigin_tl
236   {
237     \cs_if_exist_use:NTF \pdfvariable { horigin }
238     { \cs_if_exist_use:NF \pdfhorigin { 1in } }
239   }
240   \tl_const:Nx \c__shipout_vorigin_tl
241   {
242     \cs_if_exist_use:NTF \pdfvariable { vorigin }
243     { \cs_if_exist_use:NF \pdfvorigin { 1in } }
244   }

```

After the constants have been set there is no need to execute this command again, in fact it would raise an error, so we redefine it to do nothing.

```

245   \cs_gset_eq:NN \__shipout_init_page_origins: \prg_do_nothing:
246 }

```

(End definition for `__shipout_init_page_origins:`, `\c__shipout_horigin_tl`, and `\c__shipout_vorigin_tl`.)

`__shipout_picture_overlay:n` Put the argument into a `picture` environment that doesn't take up any size and uses `1pt` for `\unitlength`.

Todo: Could perhaps be generalized as it might be useful elsewhere. For now it is not.

```

247 \cs_new:Npn \__shipout_picture_overlay:n #1 {

```

The very first time this is executed we have ot initializes (and freeze) the origins.

```

248   \__shipout_init_page_origins:
249   \kern -\c__shipout_horigin_tl \scan_stop:
250   \vbox_to_zero:n {
251     \kern -\c__shipout_vorigin_tl \scan_stop:
252     \unitlength 1pt \scan_stop:

```

This mimics a simple zero-sized `picture` environment. The `\hss` is need in case there is horizontal material (without using `\put` with a positive width.

```

253     \hbox_set_to_wd:Nnn \l__shipout_tmp_box \c_zero_dim
254     { \ignorespaces #1 \hss }
255     \box_set_ht:Nn \l__shipout_tmp_box \c_zero_dim
256     \box_set_dp:Nn \l__shipout_tmp_box \c_zero_dim
257     \box_use:N \l__shipout_tmp_box
258     \tex_vss:D
259   }
260 }

```

(End definition for `__shipout_picture_overlay:n`.)

`__shipout_add_background_picture:n` Put a `picture` env in the background of the shipout box with its reference point in the top-left corner.

```

261 \cs_new:Npn \__shipout_add_background_picture:n #1 {
262   \__shipout_add_background_box:n { \__shipout_picture_overlay:n {#1} }
263 }

```

(End definition for `_shipout_add_background_picture:n`.)

`_shipout_add_foreground_picture:n` Put a `picture` env in the foreground of the shipout box with its reference point in the top-left corner.

```
264 \cs_new:Npn \_shipout_add_foreground_picture:n #1 {
265   \_shipout_add_foreground_box:n { \_shipout_picture_overlay:n {#1} }
266 }
```

(End definition for `_shipout_add_foreground_picture:n`.)

`\shipout_discard:` Request that the next shipout box should be discarded. At the moment this is just setting a boolean, but we may want to augment this behavior that the position of the call is taken into account (in case L^AT_EX looks ahead and is not using the position for on the next page).

```
267 \cs_new:Npn \shipout_discard: {
268   \bool_gset_true:N \g_shipout_discard_bool
269 }
```

(End definition for `\shipout_discard:`. This function is documented on page ??.)

3.2 Handling the end of job hook

At the moment this is partly solved by using the existing hooks. But rather than putting the code into these hooks it should be moved to the right place directly as we shouldn't prefill hooks with material unless it needs to interact with other code.

`\g_shipout_readonly_int` We count every shipout activity that makes a page (but not those that are discarded) in order to know how many pages got produced.

```
270 \int_new:N \g_shipout_readonly_int
```

For L^AT_EX 2_ε it is available as a command (i.e., a T_EX counter only).

```
271 \cs_new_eq:NN \ReadonlyShipoutCounter \g_shipout_readonly_int
```

(End definition for `\g_shipout_readonly_int` and `\ReadonlyShipoutCounter`. These functions are documented on page 5.)

`\g_shipout_totalpages_int` We count every shipout attempt (even those that are discarded) in tis counter. It is not used in the code but may get used in user code.

`\c@totalpages`

```
272 \int_new:N \g_shipout_totalpages_int
```

For L^AT_EX 2_ε this is offered as a L^AT_EX counter so can be easily typeset inside the output routine to display things like “`\thepage/\thetotalpages`”, etc.

```
273 \cs_new_eq:NN \c@totalpages \g_shipout_totalpages_int
```

```
274 \cs_new:Npn \thetotalpages { \arabic{totalpages} }
```

(End definition for `\g_shipout_totalpages_int` and `\c@totalpages`. These functions are documented on page 5.)

`\@abspage@last` In `\@abspage@last` record the number of pages from the last run. This is written to the `.aux` and this way made available to the next run. In case there is no `.aux` file or the statement is missing from it we initialize it with the largest possible number in T_EX. We use this as the default because then we are inserting the `shipout/lastpage` on the last page (or after the last page but not on page 1 for a multipage document).

```
275 \xdef\@abspage@last{\number\maxdimen}
```

(End definition for \@abspage@last. This function is documented on page ??.)

`\enddocument` Instead of using the hooks `enddocument` and `enddocument/afterlastpage` we add this code to private kernel hooks to be 100% when it is executed and to avoid cluttering the hooks with data that is always there.

Inside `\enddocument` there is a `\clearpage`. Just before that we execute this code here. There is a good change that we are on the last page. Therefore, if we don't know the value from the last run, we assume that the current page is the right one. So we set `\@abspage@last` and as a result the next shipout will run the `shipout/lastpage` code. Of course, if there are floats that still need a placement this guess will be wrong but then rerunning the document will give us the correct value next time around.

```
\@kernel@after@enddocument 276 \g@addto@macro \@kernel@after@enddocument {
277   \int_compare:nNt \@abspage@last = \maxdimen
278   {
```

We use L^AT_EX 2_ε coding as `\@abspage@last` is not an L3 name.

```
279     \xdef\@abspage@last{ \int_eval:n {\g_shipout_readonly_int + 1} }
280   }
281 }
```

Once the `\clearpage` has done its work inside `\enddocument` we know for sure how many pages this document has, so we record that in the `.aux` file for the next run.

```
\@kernel@after@enddocument@afterlastpage
```

```
282 \g@addto@macro \@kernel@after@enddocument@afterlastpage {
```

There is one special case: If no output is produced then there is no point in a) recording the number as 0 will never match the page number of a real page and b) adding an extra page to run the `shipout/lastpage` is pointless as well (as it would remain forever). So we test for this and run the code only if there have been pages.

```
283   \int_compare:nNf \g_shipout_readonly_int = 0
284   {
```

This ends up in the `.aux` so we use L^AT_EX 2_ε names here.

Todo: This needs an interface for \nofiles in expl3, doesn't at the moment!

```
285   \if@filesw
286     \iow_now:Nx \@auxout {
287       \gdef\string\@abspage@last {\int_use:N \g_shipout_readonly_int}}
288   \fi
```

But we may have guessed wrongly earlier and we still have to run the `shipout/lastpage` even though there is no page to place it into. If that is the case we make a trivial extra page and put it there. This temporary page will then vanish again on the next run but helps to keep pdf viewers happy.

```
289     \bool_if:Nf \g__shipout_lastpage_handled_bool
290     {
```

However, making this extra page in case the hook is actually empty would be forcing a rerun without any reason, so we check that condition and also check if `\@kernel@after@shipout@lastpage` contains any code. If both are empty we omit the page generation.

```
291       \bool_lazy_and:nnF
292       { \hook_if_empty_p:n {shipout/lastpage} }
```

```

293         { \tl_if_empty_p:N \@kernel@after@shipout@lastpage }
294         {
295             \tex_shipout:D\vbox to\textheight
296             {
297                 \hbox:n { \UseHook{shipout/lastpage}
298                     \@kernel@after@shipout@lastpage }

```

This extra page could be totally empty except for the hook content, but to help the user understanding why it is there we put some text into it.

```

299             \__shipout_excuse_extra_page:
300             \null
301         }

```

At this point we also signal to L^AT_EX's endgame that a rerun is necessary so that an appropriate message can be shown on the terminal. We do this by simply defining a command used as a flag and tested `\enddocument`.

```

302             \cs_gset_eq:NN \@extra@page@added \relax
303         }
304     }
305 }
306 }

```

(End definition for `\enddocument`, `\@kernel@after@enddocument`, and `\@kernel@after@enddocument@afterlastpage`. These functions are documented on page ??.)

`__shipout_excuse_extra_page:` Say mea culpa ...

```

307 \cs_new:Npn \__shipout_excuse_extra_page: {
308     \vfil
309     \begin{center}
310         \bfseries Temporary~ page!
311     \end{center}
312     \LaTeX{}~ was~ unable~ to~ guess~ the~ total~ number~ of~ pages~
313     correctly.~ ~ As~ there~ was~ some~ unprocessed~ data~ that~
314     should~ have~ been~ added~ to~ the~ final~ page~ this~ extra~
315     page~ has~ been~ added~ to~ receive~ it.
316     \par
317     If~ you~ rerun~ the~ document~ (without~ altering~ it)~ this~
318     surplus~ page~ will~ go~ away,~ because~ \LaTeX{}~ now~ knows~
319     how~ many~ pages~ to~ expect~ for~ this~ document.
320     \vfil
321 }

```

(End definition for `__shipout_excuse_extra_page:.`)

`\PreviousTotalPages` In the preamble before the aux file was read `\PreviousTotalPages` is always zero.

`\@kernel@before@begindocument` 322 `\def\PreviousTotalPages{0}`

In the aux file there should be an update for `\@abspage@last` recording the number of pages from the previous run. If not that macro holds the value of `\maxdimen`. So we test for it and update `\PreviousTotalPages` if there was a real value. This should happen just before the `begindocument` hook is executed so that the value can be used inside that hook.

```

323 \g@addto@macro\@kernel@before@begindocument
324     {\ifnum\@abspage@last<\maxdimen
325         \xdef\PreviousTotalPages{\@abspage@last}\fi}

```

(End definition for `\PreviousTotalPages` and `\@kernel@before@begindocument`. These functions are documented on page 5.)

4 Legacy L^AT_εX interfaces

`\DiscardShipoutBox` Request that the next shipout box is to be discarded.

```
326 \cs_new_eq:NN \DiscardShipoutBox \shipout_discard:
```

(End definition for `\DiscardShipoutBox`. This function is documented on page 4.)

`\AtBeginDvi`

```
327 \DeclareRobustCommand \AtBeginDvi {\_shipout_add_firstpage_material:Nn \AtBeginDvi}
```

(End definition for `\AtBeginDvi`. This function is documented on page 4.)

`\DebugShipoutsOn`

`\DebugShipoutsOff`

```
328 \cs_new_eq:NN \DebugShipoutsOn \shipout_debug_on:
```

```
329 \cs_new_eq:NN \DebugShipoutsOff \shipout_debug_off:
```

(End definition for `\DebugShipoutsOn` and `\DebugShipoutsOff`. These functions are documented on page 5.)

5 Internal commands needed elsewhere

These internal commands use double and triple @ signs so we need to stop getting them translated to the module name.

```
330 <@@=>
```

Some internals needed elsewhere.

```
\@expl@@@shipout@add@firstpage@material@@Nn
\@expl@@@shipout@add@background@box@@n
\@expl@@@shipout@add@foreground@box@@n
\@expl@@@shipout@add@background@picture@@n
\@expl@@@shipout@add@foreground@picture@@n
```

```
331 \cs_set_eq:NN \@expl@@@shipout@add@firstpage@material@@Nn
```

```
332 \_shipout_add_firstpage_material:Nn
```

```
333 \cs_set_eq:NN \@expl@@@shipout@add@background@box@@n
```

```
334 \_shipout_add_background_box:n
```

```
335 \cs_set_eq:NN \@expl@@@shipout@add@foreground@box@@n
```

```
336 \_shipout_add_foreground_box:n
```

```
337 \cs_set_eq:NN \@expl@@@shipout@add@background@picture@@n
```

```
338 \_shipout_add_background_picture:n
```

```
339 \cs_set_eq:NN \@expl@@@shipout@add@foreground@picture@@n
```

```
340 \_shipout_add_foreground_picture:n
```

(End definition for `\@expl@@@shipout@add@firstpage@material@@Nn` and others. These functions are documented on page ??.)

```
341 \ExplSyntaxOff
```

```
342 </2ekernel | latexrelease>
```

```
343 <latexrelease>\EndIncludeInRelease
```

Rolling back here doesn't undefine the interface commands as they may be used in packages without rollback functionality. So we just make them do nothing which may or may not work depending on the code usage.

```

344 <latexrelease>\IncludeInRelease{0000/00/00}%
345 <latexrelease>          {\shipout}{The hook management (shipout)}%
346 <latexrelease>\expandafter\let\expandafter\shipout\csname tex_shipout:D\endcsname
347 <latexrelease>
348 <latexrelease>\let \ShipoutBox \@undefined
349 <latexrelease>\let \ReadOnlyShipoutCounter \@undefined
350 <latexrelease>\let \c@totalpages \@undefined
351 <latexrelease>\let \thetotalpages \@undefined
352 <latexrelease>
353 <latexrelease>\let \DiscardShipoutBox \@undefined
354 <latexrelease>\let \DebugShipoutsOn \@undefined
355 <latexrelease>\let \DebugShipoutsOff \@undefined
356 <latexrelease>
357 <latexrelease>\DeclareRobustCommand \AtBeginDvi [1]{%
358 <latexrelease> \global \setbox \@begindvibox
359 <latexrelease> \vbox{\unvbox \@begindvibox #1}%
360 <latexrelease>}
361 <latexrelease>
362 <latexrelease>\let \AtBeginShipout \@undefined
363 <latexrelease>\let \AtBeginShipoutNext \@undefined
364 <latexrelease>
365 <latexrelease>\let \AtBeginShipoutFirst \@undefined
366 <latexrelease>
367 <latexrelease>\let \ShipoutBoxHeight \@undefined
368 <latexrelease>\let \ShipoutBoxDepth \@undefined
369 <latexrelease>\let \ShipoutBoxWidth \@undefined
370 <latexrelease>
371 <latexrelease>\let \AtBeginShipoutDiscard \@undefined
372 <latexrelease>
373 <latexrelease>\let \AtBeginShipoutAddToBox \@undefined
374 <latexrelease>\let \AtBeginShipoutAddToBoxForeground \@undefined
375 <latexrelease>\let \AtBeginShipoutUpperLeft \@undefined
376 <latexrelease>\let \AtBeginShipoutUpperLeftForeground \@undefined
377 <latexrelease>
378 <latexrelease>

```

We do not undo a substitution when rolling back. As the file support gets undone the underlying data is no longer used (and sufficiently obscure that should not interfere with existing commands) and properly removing it would mean we need to make the `\undeclare@...` and its support macros available in all earlier kernel releases which is pointless (and actually worse).

```

379 %\undeclare@file@substitution{everyshi.sty}{everyshi-ltx.sty}
380 <latexrelease>
381 <latexrelease>\let \AtEndDvi \@undefined

```

We do not reenabale a disabled package load when rolling back. As the file support gets undone the underlying data is no longer checked (and sufficiently obscure that it should not interfere with existing commands) and properly removing it would mean we need to make the `\reenable@package@load` command available in all earlier kernel releases which is pointless (and actually worse).

```

382 %\reenable@package@load{atenddvi}
383 <latexrelease>
384 <latexrelease>\EndIncludeInRelease
385 <*2ekernel>

```

6 Package emulation for compatibility

6.1 Package `atenddvi` emulation

`\AtEndDvi` This package has only one public command to simulating it is easy and actually sensible to provide as part of the kernel.

```

386 </2ekernel>
387 <*2ekernel | latexrelease>
388 <latexrelease>\IncludeInRelease{2020/10/01}%
389 <latexrelease>          {\AtEndDvi}{atenddvi emulation}%
390 \ExplSyntaxOn
391 \cs_new:Npn \AtEndDvi {\AddToHook{shipout/lastpage}}
392 \ExplSyntaxOff

```

As the package is integrate we prevent loading (no need to roll that back):

```

393 \disable@package@load{atenddvi}
394   {\PackageWarning{atenddvi}
395     {Functionality of this package is already\MessageBreak
396     provided by LaTeX.\MessageBreak\MessageBreak
397     It is there no longer necessary to load it\MessageBreak
398     and you can safely remove it.\MessageBreak
399     Found on}}
400 </2ekernel | latexrelease>
401 <latexrelease>\EndIncludeInRelease
402 <latexrelease>\IncludeInRelease{0000/00/00}%
403 <latexrelease>          {\AtEndDvi}{atenddvi emulation}%
404 <latexrelease>\let \AtEndDvi \@undefined
405 <latexrelease>\EndIncludeInRelease
406 <*2ekernel>

```

(End definition for `\AtEndDvi`. This function is documented on page 4.)

```

407 </2ekernel>

```

6.2 Package `atbegshi` emulation

```

408 <*atbegshi-ltx>
409 \ProvidesPackage{atbegshi-ltx}
410   [2020/08/17 v1.0a
411     Emulation of the original atbegshi package^Jwith kernel methods]

```

`\AtBeginShipoutBox`

```

412 \let \AtBeginShipoutBox \ShipoutBox

```

(End definition for `\AtBeginShipoutBox`. This function is documented on page 6.)

`\AtBeginShipoutInit` Compatibility only, we aren't delaying ...

```

413 \let \AtBeginShipoutInit \@empty

```

(End definition for `\AtBeginShipoutInit`. This function is documented on page 6.)

`\AtBeginShipout` Filling hooks
`\AtBeginShipoutNext` 414 `\def \AtBeginShipout {\AddToHook{shipout/before}}`
415 `\def \AtBeginShipoutNext {\AddToHookNext{shipout/before}}`

(End definition for `\AtBeginShipout` and `\AtBeginShipoutNext`. These functions are documented on page 6.)

`\AtBeginShipoutFirst` Slightly more complex as we need to know the name of the command under which the `shipout/firstpage` hook is filled.

```
416 \def \AtBeginShipoutFirst
417   {\@expl@@@shipout@add@firstpage@material@@Nn \AtBeginShipoutFirst}
```

(End definition for `\AtBeginShipoutFirst`. This function is documented on page 6.)

`\AtBeginShipoutDiscard` Just a different name.

```
418 \let \AtBeginShipoutDiscard \DiscardShipoutBox
```

(End definition for `\AtBeginShipoutDiscard`. This function is documented on page 6.)

`\AtBeginShipoutAddToBox` We don't expose them.

```
\AtBeginShipoutAddToBoxForeground 419 \let \AtBeginShipoutAddToBox
\AtBeginShipoutUpperLeft          420   \@expl@@@shipout@add@background@box@@N
\AtBeginShipoutUpperLeftForeground 421 \let \AtBeginShipoutAddToBoxForeground
422   \@expl@@@shipout@add@foreground@box@@N
423 \let \AtBeginShipoutUpperLeft
424   \@expl@@@shipout@add@background@picture@@N
425 \let \AtBeginShipoutUpperLeftForeground
426   \@expl@@@shipout@add@foreground@picture@@N
```

(End definition for `\AtBeginShipoutAddToBox` and others. These functions are documented on page 6.)

`\ShipoutBoxHeight` This is somewhat different from the original in `atbegshi` where `\ShipoutBoxHeight` etc.
`\ShipoutBoxWidth` only holds the `\the\ht<box>` value. This may have some implications in some use cases
`\ShipoutBoxDepth` and if that is a problem then it might need changing.

```
427 \ExplSyntaxOn
428 \cs_new:Npn \ShipoutBoxHeight { \dim_use:N \l_shipout_box_ht_dim }
429 \cs_new:Npn \ShipoutBoxDepth  { \dim_use:N \l_shipout_box_dp_dim }
430 \cs_new:Npn \ShipoutBoxWidth  { \dim_use:N \l_shipout_box_wd_dim }
431 \ExplSyntaxOff
```

(End definition for `\ShipoutBoxHeight`, `\ShipoutBoxWidth`, and `\ShipoutBoxDepth`. These functions are documented on page ??.)

```
432 </atbegshi-ltx>
```

If the package is requested we substitute the one above:

```
433 (*2kernel)
434 \declare@file@substitution{atbegshi.sty}{atbegshi-ltx.sty}
435 </2kernel>
```

6.3 Package everyshi emulation

```
436 <*everyshi-ltx>
437 \ProvidesPackage{everyshi-ltx}
438 [2020/08/17 v1.0a
439 Emulation of the original everyshi package^^Jwith kernel methods]
```

`\EveryShipout`
`\AtNextShipout`

This package has only two public commands so simulating it is easy:

```
440 \def \EveryShipout {\AddToHook{shipout/before}}
441 \def \AtNextShipout {\AddToHookNext{shipout/before}}
```

(End definition for \EveryShipout and \AtNextShipout. These functions are documented on page 6.)

```
442 % This is one difference between \pkg{everyshi} and the kernel
443 % implementation, the latter does not directly use box 255.
444 %
445 % For usage by ordinary users this makes no difference but of a
446 % package use complicated code together with \pkg{everyshi} and
447 % directly manipulates box 255 then this package needs updating.
448 % In most cases the updates are simple because the kernel offers
449 % hooks that makes such complicated code unnecessary.
450 %
451 % We therefore add a little file into the adjusted package
452 % \begin{macrocode}
453 %%
454 %% In normal circumstances the above emulation is sufficient and in
455 %% all known packages (we know of) that use everyshi it either works or
456 %% the packages have been adjusted.
457 %%
458 %% Code that directly manipulates box 255, however, might fail.
459 %% If that is the case look at the shipout hooks offered now as
460 %% they are normally sufficient to avoid such minpulations (or
461 %% replace box 255 with \ShipoutBox in the code.
462 %%
463 </everyshi-ltx>
```

If the package is requested we substitute the one above:

```
464 <*2kernel>
465 \declare@file@substitution{everyshi.sty}{everyshi-ltx.sty}
466 </2kernel>
```

Rather important :-)

```
467 <@@=>
```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

- A**
- `\AddToHook` . . . *5, 5, 5, 5, 114, 391, 414, 440*
 - `\AddToHookNext` *3, 5, 5, 415, 441*
 - `\Alph` *4*
 - `\arabic` *4, 274*
 - `\AtBeginDvi` *3, 5, 10, 10, 327, 357*
 - `\AtBeginShipout` *5, 362, 414*
 - `\AtBeginShipoutAddToBox` *5, 373, 419*
 - `\AtBeginShipoutAddToBoxForeground` *5, 374, 419*
 - `\AtBeginShipoutBox` *5, 412*
 - `\AtBeginShipoutDiscard` *5, 371, 418*
 - `\AtBeginShipoutFirst` *5, 365, 416*
 - `\AtBeginShipoutInit` *5, 413*
 - `\AtBeginShipoutNext` *5, 363, 414*
 - `\AtBeginShipoutUpperLeft` . . *5, 5, 375, 419*
 - `\AtBeginShipoutUpperLeftForeground` *5, 5, 376, 419*
 - `\AtEndDvi` *3, 6, 381, 386*
 - `\AtNextShipout` *5, 440*
- B**
- `\baselineskip` *146, 197*
 - `\begin` *309, 452*
 - `\bfseries` *310*
 - bool commands:
 - `\bool_gset_false:N` *15, 43, 51*
 - `\bool_gset_true:N` *10, 85, 268*
 - `\bool_if:NTF` *21, 49, 289*
 - `\bool_lazy_and:nnTF` *291*
 - `\bool_new:N` *6, 112, 127*
 - box commands:
 - `\box_dp:N` *118*
 - `\box_ht:N` *7, 117, 226*
 - `\box_if_empty:NTF` *40, 58*
 - `\box_if_horizontal:NTF` *158, 210*
 - `\box_if_vertical:NTF` *132, 183*
 - `\box_move_up:nn` *172, 226*
 - `\box_new:N` *23, 128*
 - `\box_set_dp:Nn`
 *145, 154, 171, 196, 207, 225, 256*
 - `\box_set_eq_drop:NN` *54*
 - `\box_set_ht:Nn`
 *144, 153, 170, 195, 206, 224, 255*
 - `\box_set_wd:Nn` *143, 169, 194, 223*
 - `\box_use:N` *88, 149, 174, 202, 227, 257*
 - `\box_wd:N` *119, 220, 228*
 - `\l_shipout_box` *1, 2, 7, 7, 11, 12, 23, 29, 40, 46, 54, 58, 69, 88, 131, 132, 139, 150, 153, 154, 158, 165, 175, 183, 190, 200, 206, 207, 210, 217, 219, 220, 226, 228*
- C**
- `\clearpage` *16, 16*
 - cs commands:
 - `\cs_gset:Npn` *106*
 - `\cs_gset_eq:NN` *31, 105, 245, 302*
 - `\cs_gset_protected:Npx` *20*
 - `\cs_if_exist_use:NTF` *237, 238, 242, 243*
 - `\cs_new:Npn` *33, 39, 102, 113, 116, 130, 181, 234, 247, 261, 264, 267, 274, 307, 391, 428, 429, 430*
 - `\cs_new_eq:NN`
 *7, 94, 271, 273, 326, 328, 329*
 - `\cs_new_protected:Npn` *8, 13, 18*
 - `\cs_set:Npn` *25*
 - `\cs_set_eq:NN` *24, 44, 56, 87, 91, 331, 333, 335, 337, 339*
 - `\csname` *346*
- D**
- `\DebugShipoutsOff` *4, 328, 355*
 - `\DebugShipoutsOn` *4, 328, 354*
 - `\DeclareRobustCommand` *327, 357*
 - `\def` *322, 414, 415, 416, 440, 441*
 - dim commands:
 - `\dim_new:N` *123, 124, 125, 126*
 - `\dim_set:Nn` *117, 118, 119, 120*
 - `\dim_use:N` *428, 429, 430*
 - `\c_max_dim` *137, 163, 188, 215*
 - `\c_zero_dim` *143, 144, 145, 151, 169, 170, 171, 194, 195, 196, 223, 224, 225, 253, 255, 256*
 - `\DiscardShipoutBox` . . *2, 3, 5, 326, 353, 418*
- E**
- `\end` *311*
 - `\endcsname` *346*
 - `\enddocument` *16, 16, 17, 276*
 - `\EndIncludeInRelease` . . *343, 384, 401, 405*
 - `\EveryShipout` *5, 440*
 - exp commands:
 - `\exp_not:N` *56, 87*
 - `\expandafter` *346*
 - `\ExplSyntaxOff` *341, 392, 431*

<code>\ExplSyntaxOn</code>	5, 390, 427	<code>\lineskiplimit</code>	148, 199
F		M	
<code>\fi</code>	288, 325	<code>\maxdimen</code>	17, 275, 277, 324
G		<code>\MessageBreak</code> .	60, 108, 395, 396, 397, 398
<code>\gdef</code>	287	N	
<code>\global</code>	358	<code>\nofiles</code>	16
group commands:		<code>\null</code>	300
<code>\group_begin:</code>	53	<code>\number</code>	275
<code>\group_end:</code>	55	P	
H		<code>\PackageInfo</code>	50
<code>\hbadness</code>	162, 164, 214, 216	<code>\PackageWarning</code>	41, 59, 394
<code>\hbox</code>	1, 2, 5	<code>\par</code>	316
hbox commands:		<code>\pdfhorigin</code>	238
<code>\hbox:n</code>	297	<code>\pdfvariable</code>	237, 242
<code>\hbox_set:Nn</code>	141, 167, 192, 221	<code>\pdfvorigin</code>	243
<code>\hbox_set_to_wd:Nnn</code> . . .	165, 217, 253	<code>\pkg</code>	442, 446
<code>\hbox_unpack:N</code>	175, 219	<code>\PreviousTotalPages</code>	4, 17, 17, 322
<code>\hfuzz</code>	161, 163, 213, 215	prg commands:	
hook commands:		<code>\prg_do_nothing:</code>	105, 245
<code>\hook_if_empty:nTF</code> . . .	70, 73, 79, 103	<code>\protect</code> . . .	7, 8, 9, 9, 9, 44, 56, 87, 91, 94
<code>\hook_if_empty_p:n</code>	292	<code>\ProvidesPackage</code>	409, 437
<code>\hook_new:n</code>	95, 96, 97, 98, 99	<code>\put</code>	2, 5, 14
<code>\hook_use:n</code>	47, 72, 76	R	
<code>\hss</code>	14, 254	<code>\ReadOnlyShipoutCounter</code>	4
I		<code>\ReadonlyShipoutCounter</code> . .	4, 4, 270, 349
<code>\ifnum</code>	4, 324	<code>\relax</code>	135,
<code>\ignorespaces</code>	254	136, 161, 162, 186, 187, 213, 214, 302	
<code>\IncludeInRelease</code>	3, 344, 388, 402	<code>\RequirePackage</code>	4
int commands:		<code>\Roman</code>	4
<code>\int_compare:nNnTF</code> . .	34, 78, 277, 283	S	
<code>\int_eval:n</code>	279	scan commands:	
<code>\int_gincr:N</code>	48, 64	<code>\scan_stop:</code>	249, 251, 252
<code>\int_new:N</code>	270, 272	<code>\setbox</code>	358
<code>\int_use:N</code>	4, 4, 66, 81, 287	<code>\shipout</code> 1, 1, 3, 4, 7, 7, 7, 8, 9, 31, 345, 346	
<code>\int_value:w</code>	27	shipout commands:	
<code>\c_max_int</code>	138, 164, 189, 216	<code>\l_shipout_box_dp_dim</code>	
<code>\c_zero_int</code>	52 1, 118, 121, 123, 154, 207, 429	
iow commands:		<code>\l_shipout_box_ht_dim</code>	
<code>\iow_now:Nn</code>	286	.. 1, 117, 120, 123, 153, 173, 206, 428	
K		<code>\l_shipout_box_ht_plus_dp_dim</code> . . .	
<code>\kern</code> .	12, 151, 201, 203, 220, 228, 249, 251 1, 120, 123, 139, 190, 201, 203	
L		<code>\l_shipout_box_wd_dim</code>	
<code>\LaTeX</code>	312, 318 1, 119, 123, 165, 217, 430	
<code>\let</code>	100,	<code>\shipout_debug_off:</code>	4, 7, 329
101, 346, 348, 349, 350, 351, 353,		<code>\shipout_debug_on:</code>	4, 7, 328
354, 355, 362, 363, 365, 367, 368,		<code>\shipout_discard:</code>	267, 326
369, 371, 373, 374, 375, 376, 381,		<code>\shipout_discard_box:</code>	3
404, 412, 413, 418, 419, 421, 423, 425		<code>\g_shipout_readonly_int</code>	
<code>\lineskip</code>	147, 198	.. 4, 64, 66, 78, 81, 270, 279, 283, 287	
		<code>\g_shipout_totalpage_int</code>	4
		<code>\g_shipout_totalpages_int</code> 4, 8, 48, 272	

