L'extension `afterpage` *

David Carlisle

28/10/2014

Ce fichier est maintenu par l'équipe du « LaTeX Project ».
Les rapports d'anomalie peuvent être envoyés en anglais à
`http://latex-project.org/bugs.html` (catégorie `tools`).

Cette extension implémente une commande, `\afterpage`, qui permet aux commandes placées en argument d'être développées après que la page courante ait été livrée [1].

1. Parfois, le mécanisme de positionnement de flottants de LaTeXse retrouve surchargé et chaque `figure` et `table` flottante est alors placée en fin de document. Il est possible de forcer la mise en place des flottants non encore traités en utilisant la commande `\clearpage`, mais ceci impose une fin prématurée de la page courante. Maintenant, vous pouvez saisir `\afterpage{\clearpage}` : la page courante sera remplie de texte normalement et une commande `\clearpage` forcera la mise en place des flottants dès la page suivante, avant que du texte ne soit disposé.
2. Il existe un mécanisme plus ancien pour aider au placement des flottants, à savoir l'argument optionnel `[H]` (pour *HERE !*, « ici ») ajouté à l'origine aux environnements standards de flottants par `here.sty`, et maintenant proposé par `float.sty`. Toutefois, des utilisateurs de ce `[H]` ont indiqué qu'ils ne souhaitaient pas vraiment placer le flottant « ici » : ils le voulaient juste placé « dans les parages ». Ceci peut maintenant être obtenu par
   `\afterpage{\clearpage\begin{figure}[H] ... \end{figure}}`
   Ceci garantit que la figure est en haut de la page suivante. (La commande `\clearpage` empêche toute autre figure de dériver après la figure associée au `H`.)
3. les tables longtable flottantes. L'extension longtable fournit un environnement `longtable`, une version sur plusieurs pages de `tabular`. De nombreux utilisateurs de `longtable` m'ont dit rencontrer des difficultés pour gérer le texte autour de ce type de table et qu'ils voulaient disposer d'une version

*Ce fichier a pour numéro de version v1.08 et a été mis à jour le 28/10/2014.

1. Il s'agit vraiment d'une préversion afin de voir si des personnes recherchent une telle commande. L'implémentation *n'est pas* particulièrement robuste. Elle ne fonctionne pas en mode deux colonnes et peut être « perturbée » par les environnements flottants de LaTeX.

flottante. Comme, par principe même, ces tables sont longues, elles sont probablement trop grandes pour tenir en mémoire et être gérées comme des `table` flottantes. Cependant, si la table est dans un fichier séparé, par exemple `ltable.tex`, vous pouvez utiliser une des deux lignes suivantes :
`\afterpage{\clearpage\input{ltable}}`
`\afterpage{\clearpage\input{ltable}\clearpage}`.
La première forme fait apparaître du texte sur la même page que la fin de la `longtable`, la seconde garantit que le texte encadrant commence aussi sur une nouvelle page.

```
1 ⟨*package⟩
```

`\afterpage` Le registre d'unités lexicales (*token*) utilisé pour sauvegarder l'ancienne routine de sortie.

```
2 \newtoks\AP@output
3 \global\AP@output\expandafter{\the\output}
```

Un registre de boîte utilisé pour sauvegarder toute partie de la page suivante qui a déjà été traitée.

```
4 \newbox\AP@partial
```

Un registre de boîte utilisé pour sauvegarder tout texte de note de bas de page qui est lié au texte qui est sauvegardé dans `\AP@partial`.

```
5 \newbox\AP@footins
```

La commande suivante essaye d'entrer proprement en mode vertical et d'appeler alors une routine de sortie spéciale pour placer la page courante dans `\AP@partial`.

```
6 \def\AP@savetop{%
```

Maintenant commence un test pour voir dans quel état nous nous trouvons. `\AP@noindent` va être défini afin que revenir dans cet état (enfin, presque) à la fin du traitement fait par afterpage.

```
7   \ifvmode
```

Nous sommes ici en mode vertical. C'est le cas le plus simple : il n'y a rien à faire.

```
8     \let\AP@noindent\empty
9   \else\ifhmode
```

Nous sommes en mode horizontal. Nous « repassons » en mode vertical, retranchant au passage la boîte d'indentation. S'il n'y a pas cette boîte d'indentation, cela implique que la routine de sortie a été invoquée par `\noindent` (vraiment pas de chance !), nous devons donc nous rappeler par la suite que nous aurons à réinsérer un `\noindent` pour le paragraphe à venir. Les unités lexicales `\everypar` ont déjà été insérées, il ne faut donc pas en insérer de nouveau.

```
10    \setbox\z@\lastbox
11    \edef\AP@noindent
12      {{\everypar{}\ifvoid\z@\noindent\else\indent\fi}}%
13    \par
14  \else
```

La possibilité (bien pire) qu'il reste est que la routine de sortie a été déclenchée par le début de mathématiques en mode hors texte dans un paragraphe.

Nous sortons du mode hors texte avec `$$` puis ajustons l'espacement (entrant en même temps dans le mode vertical). `\AP@noindent` relancera plus tard le mode hors texte. Les unités lexicales `\everydisplay` ont déjà été insérées (elles s'appliquent à la liste mathématique qui sera entamée par `\AP@noindent`, même si la présence de ces unités lexicales a été déclenchée par le mode mathématique hors texte qui a été fermé par les lignes ci-dessous !). Les valeurs de `\prevgraf` et de `\predisplaysize` sont sauvegarder pour être utilisées pour la future liste mathématique.

```
15    \abovedisplayshortskip\z@\abovedisplayskip\z@
16    \belowdisplayshortskip\z@\belowdisplayskip\z@
17    \xdef\AP@disp{%
18      \predisplaysize\the\predisplaysize
19      \prevgraf\the\prevgraf\relax}%
20    $$\vskip-\baselineskip\vskip-\parskip
21    \edef\AP@noindent{%
```

Ne pas insérer d'unites lexicales `\everydisplay` à nouveau.

```
22      \toks@{\the\everydisplay}\everydisplay{}%
```

Le mode hors texte commence sans une fausse ligne de paragraphe au-dessus de lui. Les valeurs `\prevgraf` et `\predisplaysize` sont restaurées. La commande `\aftergroup` est utilisée pour restaurer la bonne configuration de `\everydisplay` après que ce mode hors texte se soit achevé.

```
23      {\everypar{}\noindent}$$\AP@disp\aftergroup\noexpand\AP@ed}%
24  \fi\fi
```

Maintenant nous passons en routine de sortie et nous passons l'ensemble des éléments de la page dans la boîte `\AP@partial`.

```
25  \begingroup
26  \nointerlineskip\null
27    \output{%
28      \global\setbox\AP@partial\vbox{%
29        \unvbox\@cclv
30         \global\setbox\@ne\lastbox}%
```

Si le texte qui est sauvegardé dans `\AP@partial` a des notes de bas de page, nous les prenons aussi car sinon elles pourraient revenir sur la page précédent la page qui a la marque de note de bas de page ! Cet ajout a été fait en version V1.08.

```
31       \global\setbox\AP@footins\box\footins}%
```

Ayant défini la routine de sortie, nous la déclenchons...

```
32  \eject
33  \endgroup}
```

`\AP@` stocke toutes les commandes qui doivent être exécutées après le saut de page.

```
34 \let\AP@\relax
```

Le registre `\everydisplay` est restauré. La commande `\ignorespaces` empêche d'avoir un espace ou une nouvelle ligne après le `$$` créant une indentation ou un paragraphe scélérat.

```
\def\AP@ed{\everydisplay\expandafter{\the\toks@}\ignorespaces}
```

La liste verticale actuelle est retranchée, les commandes `\AP@` sont insérées en haut de la page et le texte sauvegardé est inséré de nouveau.

```
\def\AP@@{%
  \AP@savetop
  \global\expandafter\let\expandafter\AP@\expandafter\relax\AP@
  \par
```

Le texte à l'origine en haut de cette page est maintenant stocké dans la boîte `\AP@partial`, ressort `\topskip` compris. Maintenant nous voulons ✖ **sortir de la boîte** ✖ `\AP@partial`, plaçant la ligne de base de la première rangée à une distance `\baselineskip` sous la ligne de base de la dernière ligne veant du texte de la page d'après. Si nous avions supposé que rien n'est trop haut ou trop profond (et que `\topskip` est rigide), il serait alors trivial de positionner le contenu de `\AP@partial` afin que la ligne de la base de la première rangée soit à une distance `\baselineskip` de la dernière rangée tout juste ajoutée.

Dans cette version, j'ai pensé qu'il pourrait être amusant d'essayer de répliquer exactement le calcul de `\baselineskip`–ou–`\lineskip` que TEX fait normalement en interne. Le recours à `\addboxcontents` fait ce qui est souhaité (je l'espère).

```
  \addboxcontents\AP@partial
```

Maintenant le texte des notes de bas de page est inséré à nouveau. Ceci ne pourrait pas être le bon endroit pour le faire, dans la mesure où le texte vient d'être sorti de la boîte pourrait générer un saut de page du fait de son nouvel emplacement. De plus, la note pourrait ne pas avoir le bon numéro si le texte de `\afterpage` contient lui-même des notes de bas de page. Quel dommage !

```
  \ifvoid\AP@footins\else
        \insert\footins{\unvbox\AP@footins}\fi
```

Maintenant les choses sont réparées ✖ **comme si** ✖ nous avions démarré en mode horizontal.

```
  \AP@noindent}
```

Si `\AP@` n'est pas équivalente à `\relax` alors la page actuelle a déjà des commandes « afterpage », alors nous ajoutons les nouvelles commandes à la fin de la liste. Sinon, nous sauvegardons les commandes dans `\AP@` (au sein du groupe local) et nous changeons de routine de sortie (la nouvelle appelle juste l'ancienne si elle est appelée par un flottant LATEX).

```
\long\def\afterpage#1{%
  \ifx\AP@\relax
    \gdef\AP@{{#1\par}}%
    \global\output{%
      \the\AP@output
      \ifnum\outputpenalty>-\@Mi
        \global\output\expandafter{\the\AP@output}%
```

```
        \aftergroup\AP@@
      \fi}%
  \else
    \expandafter\gdef\expandafter\AP@\expandafter{\AP@{#1\par}}%
  \fi}
```

Si nous sommes arrivés à la fin de document ou à une page vide, nous déposons les éléments sans aucune bidouille.

```
\let\AP@clearpage\clearpage
\def\clearpage{%
  \ifx\AP@\relax
    \AP@clearpage
  \else
    \global\output\expandafter{\the\AP@output}%
    \AP@clearpage
```

À ce point (depuis la version v1.08), nous devons nettoyer `\AP@` *avant* de la développer car sinon elle produit une boucle infini. Soupir...

```
    \global\expandafter\let\expandafter\AP@\expandafter\relax
    \expandafter\expandafter\AP@
  \fi}
\let\AP@enddocument\enddocument
\def\enddocument{%
  \ifx\AP@\relax\else
    \global\output\expandafter{\the\AP@output}%
    \AP@clearpage
    \global\expandafter\let\expandafter\AP@\expandafter\relax
    \expandafter\expandafter\AP@
  \fi
    \AP@enddocument}
```

`\addboxcontents` Etant donné une boîte verticale `#1`, nous l'ajoutons à la liste verticale courante de façon à ce que le résultat soit équivalent à la liste que TEX aurait produit si le contenu de `#1` (hors tout ressort initial) avait été ajouté individuellement à la liste courante.

Ainsi, le problème est essentiellement celui de retirer le contenu de la boîte `#1`, ✖ **tout en** ✖ remplaçant le ressort en haut de `#1` avec le ressort de `\baselineskip` ou `\lineskip` (du moins quelque chose d'équivalent) que TEX aurait normalement placé avant la première boîte dans `#1`. De plus, la commande `\prevdepth` doit être définie à la fin.

```
\def\addboxcontents#1{{%
```

Peut-être ne devrais-je pas utiliser de groupement ici car je ne veux pas vraiment enregistrer `#1`. Si il est retranché, `\splittopskip` et `\splitmaxdepth` auraient besoin d'être restaurés manuellement.

Tout d'abord, le ressort en haut est remplacé par `\vskip 0pt`.

```
  \splittopskip\z@
  \splitmaxdepth\maxdimen
  \setbox#1\vbox{\break\unvbox#1}%
  \setbox\z@\vsplit#1to\z@
```

Le point de coupure est repositionné.

```
\setbox#1\vbox{\break\unvbox#1}%
```

La commande `\skip@` est définie comme étant la hauteur de `#1` (sans le ressort du haut).

```
\skip@\ht#1%
```

Maintenant, la première ligne de base de la première rangée est placée à `\vsize` du haut (ceci suppose que la première rangée a une hauteur inférieure à `\vsize`.)

```
\splittopskip\vsize
\setbox\z@\vsplit#1to\z@
```

La nouvelle hauteur de `#1` est retranchée de `\skip@` et ajoutée en retour à `\splittopskip` de façon à ce que `\skip@` soit maintenant la hauteur de la première rangée de `#1`. Celle-ci peut être de 0pt si (par exemple) une commande de marque (ou que sais-je encore) est placée entre le ressort du haut et la première boîte. Cette « dimension − `\splittopskip` » est enregistrée dans `\skip\tw@`.

```
\advance\skip@-\ht#1%
\skip\tw@\skip@
\advance\skip@\splittopskip
```

Maintenant, le calcul simulé de TEX est effectué.

```
\advance\skip@\prevdepth
\advance\skip@-\baselineskip
\advance\skip\tw@\ifdim-\skip@<\lineskiplimit\lineskip\else-\skip@\fi
```

Finalement, le ressort est ajouté.

```
\vskip\skip\tw@
```

Maintenant le contenu de la boîte est sorti de cette dernière, tout en définissant manuellement `\prevdepth` car `\unvbox` (contrairement à `\box`) ne le fait pas automatiquement.

```
\global\dimen@i\dp#1%
\unvbox#1}%
\prevdepth\dimen@i}
```

⟨/package⟩