

Update marks in box*

WenJian Chern[†]

September 6, 2023 v0.2c

Abstract

There are some cases where one may put title in box and still want its `\markboth` working. This brings the `updatemarks` package. The `updatemarks` package provides interface to extract marks where are in a inner box and then can put the first and last marks back to outer. It can automatically update marks where are in `minipage`, *boxed multicols* and `tcolorbox` environments.

Contents

1	Usage of the package	1
2	<code>minipage</code>	2
3	<code>multicol</code> and <code>adjmulticol</code>	2
4	<code>tcolorbox</code>	2
5	Set automatical updating list	2
6	Generic interfaces of extracting and updating	3
7	Disable patches or write your own patches	3
8	Programming interfaces	4

1 Usage of the package

You simply insert `\usepackage{updatemarks}` or `\usepackage[<options>]{updatemarks}` in the preamble of your document.

There are three package options `minipage`, `multicol`(an optional `s`) and `tcolorbox`, which are used to enable the function of automatical updating marks in `minipage`, `multicols` and `tcolorbox` (both `tcolorbox` environment and `\tcbox` command), respectively.

No automatical updatings are enable, by default.

If your L^AT_EX version is 2022-06-01 or newer, you can use `\SetKeys[updatemarks]{<options>}` to enable or disable automatical updating locally.

Such as,

```
\SetKeys[updatemarks]{minipage=false, tcolorbox=true, multicol}
```

so marks in `minipage` would not update, whereas `tcolorbox` and `multicols` do.

The total empty mark will not be extracted and updated. But `\markboth{}{}` and `\InsertMark{}` do will be extracted and updated, because they are not total empty internally.

However, the updating is not once for all. For example, if you are using `minipage` in `\makebox`, even if you enable the automatical updating, the marks still can not be found by

*Issues: <https://github.com/Sophanatprime/updatemarks/issues>

[†]Email: longaster@163.com

L^AT_EX. But, if you put `minipage` in `tcolorbox` or the other way round, L^AT_EX still can get correct marks.

The point is, updating marks is done inner to outer and level by level, if any level is not updated, you will lost all marks in inner boxes. Furthermore, whether marks are updated or not is only influenced by box level instead of group level, so if you try to limit marks in a certain position, you should use a box instead of `\begingroup` and `\endgroup`.

The extracting and updating would not remove these marks, they still be there, you can find them by your own method as well.

2 minipage

You can enable automatically extracting and updating marks where are in `minipage` to its nearest outer box level, by using `minipage` package option or set `minipage` key to true.

`updatemarks` patches the `\endminipage`, if you enable the function, marks in any other environments that use `minipage` will be updated, including `varwidth` of `varwidth` package, etc.

3 multicol and adjmulticol

If `multicols` or `adjmulticols` or their starred versions are put in a box — so called *boxed multicols*, you are able to enable automatically extracting and updating marks into the box (not outer of the box, because they are only updated one level). If you need to update marks in this box, you have to do it manually. Of cause, if the box is `minipage` or `tcolorbox` or another `multicols` or any other supported environments or commands, then no more things to do.

All `multicols` and `adjmulticols` and their starred versions share the same option `multicol`(an optional `s`).

For *non boxed multicols*, i.e, not in any other box except the main vertical list, the `\topmark` (and `\topmarks`) at first and last page are not correct. Multiple contiguous forced break or `\clearpage` may cause the wrong marks.

If your L^AT_EX version is 2022-06-01 or newer, you can get more correct mark values by using `\TopMark`, `\FirstMark` and `\LastMark` in head and foot, but only the `previous-page` region and `page` region are supported for now. Still, Multiple contiguous forced break or `\clearpage` may cause the wrong marks.

4 tcolorbox

The `updatemarks` package can also update marks in `tcolorbox` environments (both breakable and unbreakable) and `\tcbox` commands.

You are allowed to use `updatemarks=true` or `updatemarks` in `\tcboxset` or as environment and command option to enable the function, and `updatemarks=false` to disable.

5 Set automatical updating list

`updatemarks` can automatically detect mark classes allocated by `\newmarks` and `\NewMarkClass` in preamble. But if you allocate mark class after preamble (which I strongly recommend you do not), `updatemarks` also provides interfaces to enable you add these mark class to automatical updating list. If you are not using these two commands, then nothing need to be done.

<code>\AddToUpdateMarksList</code>	<code>\AddToUpdateMarksList</code>	<code>{\langle number list \rangle}</code>
<code>\SetUpdateMarksList</code>	<code>\SetUpdateMarksList</code>	<code>{\langle number list \rangle}</code>
<code>\RemoveFromUpdateMarksList</code>	<code>\RemoveFromUpdateMarksList</code>	<code>{\langle number list \rangle}</code>

You can use these functions to globally add items to, set items, and remove items from the list of mark classes which need to be automatically updated.

The $\langle number \rangle$ is the first argument of `\newmarks`, or literally, a number. Specially, for those mark classes declared by `\NewMarkClass`, you are able to use `[\langle class \rangle]` or `\MarkClass{\langle class \rangle}`. Such as

```
\RemoveFromUpdateMarksList { [2e-right], 4, \MarkClass{my-class} }
```

<code>\AddAllocatedToUpdateMarksList</code>	<code>\AddAllocatedToUpdateMarksList</code>
---	---

The function add all mark classes allocated by `\newmarks` and `\NewMarkClass` to automatic updating list.

6 Generic interfaces of extracting and updating

If you need to extract and update marks manually, `\ExtractMarks`, `\ExtractSplitMarks` and `\UpdateMarks` will help you.

<code>\ExtractMarks</code>	<code>\ExtractMarks</code>	<code>{\langle box number \rangle}</code>
	<code>\ExtractMarks</code>	<code>[\langle number list \rangle] {\langle box number \rangle}</code>
	<code>\ExtractMarks *</code>	<code>{\langle text \rangle}</code>
	<code>\ExtractMarks *</code>	<code>[\langle number list \rangle] {\langle text \rangle}</code>

Save first marks and last marks of specified mark classes which are directly presented in $\langle box number \rangle$ or $\langle text \rangle$ if these marks is not total empty. Marks in deeper boxes will not be detected, unless they are moved out.

If $\langle number list \rangle$ is presented, the mark classes are these numbers, otherwise they are the automatical updating list.

The $\langle box number \rangle$ is the first argument of `\newbox` or `\newsavebox`, etc., and have saved contents by using `\lrbox` or `\sbox` or similar.

The $\langle text \rangle$ is typeset material and *will be executed*.

If the $\langle box number \rangle$ is a vbox and contains forced page break, then all marks after the first forced page break will not be detected.

<code>\ExtractSplitMarks</code>	<code>\ExtractSplitMarks</code>
	<code>\ExtractSplitMarks</code> [<i>\langle number list \rangle</i>]

This command is used after a `\vsplit`, and save first marks and last marks of specified mark classes which are directly presented in being split part if these marks is not total empty. Marks in deeper boxes will not be detected, unless they are moved out.

If $\langle number list \rangle$ is presented, the mark classes are these numbers, otherwise they are the automatical updating list.

<code>\UpdateMarks</code>	<code>\UpdateMarks</code>
<code>\UpdateMarks</code>	<code>\UpdateMarks</code>
	<code>[\langle number list \rangle]</code>

Reinserting saved first marks and last marks of specified mark classes (if have been saved) into the current box, or if not in a box then the main vertical list.

If $\langle number list \rangle$ is presented, the mark classes are these numbers, otherwise they are the automatical updating list.

7 Disable patches or write your own patches

By default, `updatemarks` use its own patches to support `minipage`, `multicols` and `tcolorbox`, however, you can write your own patches and remove the patches will be done by `updatemarks`.

To the patches for `minipage`, you can define `\updatemarks@minipage@patch` before `updatemarks` is loaded, then `updatemarks` will use your patches. Specially, if you set `\updatemarks@minipage@patch` to empty, then no patches will be done for `\endminipage`.

The patches for `multicol`, `adjmulticol` and `tcolorbox`, are `\updatemarks@multicol@patch`, `\updatemarks@adjmulticol@patch`, `\updatemarks@tcolorbox@patch` and `\updatemarks@multicolnewmark@patch`, `\updatemarks@adjmulticolnewmark@patch` for new mark mechanism, you are able set them in preamble before or after `updatemarks` is loaded.

8 Programming interfaces

This section describes the interfaces of L^AT_EX3.

<code>\updatemarks_extract:nN</code>	<code>\updatemarks_extract:nN {<box material>} {<seq var>}</code>
<code>\updatemarks_extract_split:N</code>	<code>\updatemarks_extract_split:N {<seq var>}</code>

These two functions have the same functions of `\ExtractMarks` and `\ExtractSplitMarks`.

`<box material>` is content to build a box. Such as unpacked box using `\hbox_unpack:N` or `\vbox_unpack:N` or text.

`<seq var>` is the number sequence of mark classes which are need to be extracted.

They only save the marks whose positions are first or last.

<code>\updatemarks_update:N</code>	<code>\updatemarks_update:N {<seq var>}</code>
------------------------------------	--

This function has the same function of `\UpdateMarks`.

`<seq var>` is the number sequence of mark classes which are need to be reinserted into the current box or the main vertical list.

It only reinserts marks whose positions are first or last.

<code>\updatemarks_save:Nnn</code>	<code>\updatemarks_save:Nnn {<seq var>} {<position>} {<value code>}</code>
------------------------------------	--

Saving marks at `<position>` of specified mark classes `<seq var>`, whose values are expanding `<value code>` once.

If `\l_updatemarks_nonempty_bool` is set to `true`, then the expanded value will not be saved if it is total empty.

The `<value code>` receives each item in the `<seq var>` as a trailing brace group.

<code>\updatemarks_save_x:Nnn</code>	<code>\updatemarks_save_x:Nnn {<seq var>} {<position>} {<value code>}</code>
--------------------------------------	--

Saving marks at `<position>` of specified mark classes `<seq var>`, whose values are fully expanding `<value code>`.

If `\l_updatemarks_nonempty_bool` is set to `true`, then the expanded value will not be saved if it is total empty.

The `<value code>` receives each item in the `<seq var>` as a trailing brace group.

<code>\updatemarks_alias:Nnn</code>	<code>\updatemarks_alias:Nnn {<seq var>} {<alias position>} {<source position>}</code>
-------------------------------------	--

Setting marks at `<alias position>` are equal to `<source position>` of specified mark classes `<seq var>`.

If `\l_updatemarks_nonempty_bool` is set to `true`, then the expanded value will not be saved if it is total empty.

<code>\updatemarks_remove:Nn</code>	<code>\updatemarks_remove:Nn {<seq var>} {<position>}</code>
-------------------------------------	--

Remove marks at `<position>` of specified mark classes `<seq var>`.

<code>\updatemarks_value:nn *</code>	<code>\updatemarks_value:nn {<position>} {<mark class>}</code>
--------------------------------------	--

Return the value of `<mark class>` at `<position>`. If it has not been saved, then return to total empty.

T_EXhackers note: The result is returned within the `\unexpanded` primitive (`\exp_not:n`).

<code>\g_updatemarks_max_int</code>	Readonly interger, its value is the maximum mark class number allocated.
-------------------------------------	--

<code>\g_updatemarks_seq</code>	The number sequence which holds the mark classes needed to be automatical updated.
---------------------------------	--

<code>\g_updatemarks_classes_seq</code>	Readonly number sequence which holds the mark classes allocated by <code>\NewMarkClass</code> . If <code>\NewMarkClass</code> is undefined, then it's empty.
---	--

<code>\l_updatemarks_nonempty_bool</code>	A local <code>bool</code> variable which is used to control if is going to save a total empty value.
---	--
