

# Das mpgmpar-Package – Randnotizen auch in Minipages\*

Paul Ebermann<sup>†</sup>

27. Juli 2006

## Zusammenfassung

Innerhalb von Boxen wie etwa einer `minipage`-Umgebung sind bekanntlich `\marginpar`-Befehle nicht erlaubt – dieses Package hat eine (Teil-)Lösung.

## Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
<b>2</b>	<b>Benutzerdoku</b>	<b>1</b>
2.1	Probleme/Nachteile . . . . .	2
<b>3</b>	<b>Implementation</b>	<b>2</b>
3.1	Vorbereitungen . . . . .	2
3.2	Interne Kommandos . . . . .	3
3.3	Neue <code>minipage</code> -Umgebung . . . . .	4
3.4	Ende . . . . .	5
<b>4</b>	<b>Liste der Änderungen</b>	<b>5</b>
<b>5</b>	<b>Index</b>	<b>5</b>

## 1 Einleitung

Innerhalb von Boxen wie etwa einer `minipage`-Umgebung sind `\marginpar`-Befehle nicht erlaubt, das heißt, sie bewirken nichts außer einer Fehlermeldung („Float(s) lost“).

Das Paket `marginnote`<sup>1</sup> von Markus Kohm umgeht das, indem nicht-gleitende Marginalien bereitgestellt werden.

Dieses Paket verfolgt einen anderen Ansatz: Es wird ein Mechanismus bereitgestellt, mit dem die `\marginpar`-Befehle abgefangen und dann außerhalb dieser Box ausgeführt werden können.

---

\*Dieses Dokument gehört zu `mpgmpar` v0.2, vom 2006/05/23.

<sup>†</sup>`Paul-Ebermann@gmx.de`

<sup>1</sup>auf CTAN unter `macros/latex/contrib/marginnote/`

## 2 Benutzerdoku

`minipagewithmarginpars` Diese Umgebung funktioniert wie die `minipage`-Umgebung aus dem L<sup>A</sup>T<sub>E</sub>X-Kernel, mit dem Unterschied, dass in ihr vorkommende `\marginpar`-Befehle erst am Ende der Umgebung, nach der Minipage selbst, ausgeführt werden.

Hier ein Beispiel:

```

Text am Anfang.      \begin{minipagewithmarginpars}{4cm}
                      Text am Anfang\marginpar{Bla}.

                      \vspace{1.5cm}
                      Text am Ende\marginpar[links]{rechts}.
Text am Ende.        \end{minipagewithmarginpars}

```

Die Randnotizen werden in der durch die (hier zwei, der Code ist auch eine) Minipages gebildeten Zeile abgesetzt – dabei ist die Default-Ausrichtung hier `[t]` anstatt `[c]`.

### 2.1 Probleme/Nachteile

- Die Marginalien werden alle ab der selben Zeile (jeweils etwas nach unten verschoben) gesetzt, nicht entsprechend der Zeile (in der Minipage), in der der `\marginpar`-Befehl vorkam.

Dies ist prinzipbedingt, ich habe keine Idee, was man dagegen tun könnte. (Je nach Problem gibt `marginnote` wohl bessere Ergebnisse.)

- Wird die `minipagewithmarginpars` innerhalb einer weiteren Box verwendet (z.B. um noch einen Rahmen zu setzen), funktioniert es weiterhin nicht, da ja dort ebenfalls keine `\marginpars` erlaubt sind.

Hierfür gibt es eine Lösung – die gesicherten `\marginpars` müssen einfach erst nach der äußersten Box (die hoffentlich in einer horizontalen Liste ist) ausgeführt werden. Dazu kann man sich analog zu `minipagewithmarginpars` eine entsprechende Umgebung definieren, Details sind in Abschnitt 3.3 im Implementations-Teil nachzulesen.

- Befinden sich die `\marginpar`-Befehle innerhalb einer der AMS-Mathe-Umgebungen (wie etwa `align`, `gather` und ihre `*`-Varianten), so kann es vorkommen, dass die Marginalien doppelt auftauchen. Dies liegt daran, dass diese Umgebungen ihren Inhalt mehrfach auswerten – und jedesmal wird dann der Inhalt gespeichert. Dieses Problem betrifft auch andere Makros, die ähnlich vorgehen.

Ich halte dies für einen Bug (in `mpgmpar`), habe aber noch keine Idee, was ich dagegen tun könnte.

## 3 Implementation

```
1 (*package)
```

### 3.1 Vorbereitungen

`\mpgmpar@savedmargins` Dieses Makro ist einfach nur ein „Behälter“ für die aufgesparten `\marginpar`-Befehle. Wir definieren es hier (leer) mittels `\newcommand`, um bei Konflikten eine

Fehlermeldung zu erhalten.

```
2 \newcommand*{\mpgmpar@savemargins}{}%
```

`\mpgmpar@dummy` Diese Kontrollsequenz wird nur verwendet, um einen nicht vorhandenen Parameter erkennen zu können. (Wir definieren es zunächst als Makro, um Konflikte zu erkennen, lassen es nachher aber gleich `\relax` sein.)

```
3 \newcommand*{\mpgmpar@dummy}{}%
4 \let \mpgmpar@dummy = \relax
```

## 3.2 Interne Kommandos

Unsere beiden Makros `\mpgmpar@savemarginpars` (am Anfang eines Bereiches) und `\mpgmpar@restoremarginpars` (am Ende) machen die eigentliche Arbeit und können auch für die Definition eigener *Box-Making*-Umgebungen verwendet werden, für eine Anleitung dafür siehe Abschnitt 3.3.

`\mpgmpar@savemarginpars` Dieses Makro leitet einen Bereich ein, in dem `\marginpars` aufgespart werden (er geht bis zum Ende der aktuellen Gruppe).

```
5 \newcommand*{\mpgmpar@savemarginpars}{%
```

Wir definieren `\marginpar` neu.

`\marginpar` Es hat wie das Original-`\marginpar` einen optionalen und einen verpflichtenden Parameter. (Um den Fall, dass der optionale Parameter angegeben wurde, von dem Fall der Nichtangabe zu unterscheiden, nehmen wir als Default `\mpgmpar@dummy` und vergleichen nachher damit.)

```
6 \renewcommand*{\marginpar}[2][\mpgmpar@dummy]{%
7 {%
```

Das `\@bsphack` und das dazugehörige `\@esphack` am Ende sind Kernel-Kommandos, welche (zusammen) dafür sorgen, dass an der Stelle der Verwendung keine Spur des `\marginpar`-Aufrufs bleibt.

```
8 \@bsphack
```

Jetzt der Vergleich ... bis zu Version 0.1 hatte ich den mit

```
\ifthenelse{\equal{\mpgmpar@dummy}{\#1}}{\dots
```

aus `ifthen` gemacht, aber das hatte Nebenwirkungen (da für den Vergleich `\#1` expandiert wurde). Daher direkt mit `\ifx`.

```
9 \def\temp@a{\#1}%
10 \ifx\mpgmpar@dummy\temp@a%
```

Dann kopieren wir (global) einfach den `\marginpar`-Aufruf an das Ende unseres „Speicher-Makros“ `\mpgmpar@savemargins`.

```
11 \g@addto@macro{\mpgmpar@savemargins}{%
12 \marginpar{\#2}}%
13 \else%
14 \g@addto@macro{\mpgmpar@savemargins}{%
15 \marginpar[\#1]{\#2}}%
16 \fi%
17 \@ignorefalse
18 \@esphack
19 }%
20 }%
```

Das `\@bsphack-\@esphack`-Paar ist hier vorhanden, damit sich unser modifiziertes `\marginpar` bezüglich umrundender Leerzeichen o.ä. genauso verhält wie das Original-`\marginpar`.

`\mpgmpar@restoremarginpars` Dieses Makro führt die gespeicherten `\marginpar`-Befehle aus und leert dann die Liste.

```
21 \newcommand*{\mpgmpar@restoremarginpars}{%
```

Zuerst merken wir uns die Liste im Makro `\@tempa` (das ist für derartige Sachen gedacht), dann löschen wir (global) `\mpgmpar@savemargins`.

```
22 \let \mpgmpar@savemargins = \mpgmpar@savemargins
```

```
23 \global\let \mpgmpar@savemargins = \empty
```

Die eben kopierte Liste führen wir nun aus (falls wir jetzt noch in einer äußeren Umgebung mit unserem Spezial-`\marginpar` sind, wird die Liste dadurch neu angelegt, andernfalls werden die Randnotizen ausgegeben), und löschen dann die Kopie.

```
24 \@tempa
```

```
25 \let \mpgmpar@savemargins = \undefined
```

```
26 }%
```

### 3.3 Neue minipage-Umgebung

Diese Umgebung dient als Beispiel für die Erstellung derartiger Umgebungen mit Hilfe von `\mpgmpar@savemarginpars` und `\mpgmpar@restoremarginpars`. Das Wesentliche dabei ist: `\mpgmpar@savemarginpars` sollte innerhalb einer Gruppe aufgerufen werden (meist nahe am Anfang), `\mpgmpar@restoremarginpars` nach dem Ende dieser Gruppe (an der Stelle, wo die Randnotizen erscheinen sollen).

Mit etwas Eigenarbeit dürfte es auch ohne eine (weitere) Gruppe klappen. Dafür muss man am Anfang `\marginpar` mit `\let` sichern und am Ende wiederherstellen:

```
\let \savedmarginpar = \marginpar
```

```
\mpgmpar@savemarginpars
```

```
...
```

```
\let \marginpar = \savedmarginpar
```

```
\mpgmpar@restoremarginpars
```

Das ist aber nicht von mir getestet, also ohne Garantie. Und es ist eher nicht robust gegenüber Verschachtelungen derartiger Konstrukte.

`minipagewithmarginpars` Hier nun unserer neue Minipage-Umgebung. Sie hat einen optionalen (vertikale Ausrichtung – Default ist `t`) und einen verpflichtenden (Breite) Parameter.

```
27 \newenvironment*{minipagewithmarginpars}[2][t]{%
```

Die Implementation ist einfach: Wir beginnen zunächst die Original-`minipage`-Umgebung (mit den selben Parametern) und innerhalb davon rufen wir unser Makro `\mpgmpar@savemarginpars` auf.

```
28 \begin{minipage}[#1]{#2}%
```

```
29 \mpgmpar@savemarginpars
```

```
30 } {%
```

Am Ende beenden wir zunächst die Minipage (wodurch `\marginpar` seine Original-Bedeutung wiedererlangt), und rufen dann `\mpgmpar@restoremarginpars` auf.

```
31 \end{minipage}%
32 \mpgmpar@restoremarginpars
33 }%
```

### 3.4 Ende

... Das war es.

```
34 \endinput
35 \end{package}
```

## 4 Liste der Änderungen

v0.0		v0.2	
	Allgemein: Erste Fassung . . . . .	1	Allgemein: ifthen wird nicht mehr benötigt. . . . .
v0.1			2
	Allgemein: Erste veröffentlichte Fassung . . . . .	1	\marginpar: Vergleich jetzt mit \ifx statt \ifthenelse. . . . .
			3

## 5 Index

Schrägedruckte Nummern verweisen auf die Seite, auf der der Eintrag beschrieben ist, unterstrichene Nummern zeigen auf die Zeilennummer der Definition, sonstige Zahlen auf die Zeilennummer einer Verwendung.

Symbols		G	
\@bsphack . . . . .	8	\g@addto@macro . . . . .	11, 14
\@empty . . . . .	23		
\@esphack . . . . .	18	I	
\@ignorefalse . . . . .	17	\ifx . . . . .	10
D		M	
\def . . . . .	9	\marginpar . . . . .	<u>6</u>
E		minipagewithmarginpars (environ- ment) . . . . .	<u>1</u> , <u>27</u>
\else . . . . .	13	\mpgmpar@dummy . . . . .	<u>3</u> , 6, 10
environments:		\mpgmpar@restoremarginpars . . .	<u>21</u> , 32
minipagewithmarginpars . . . . .	<u>27</u>	\mpgmpar@savedmargins <u>2</u> , 11, 14, 22, 23	
environments:minipagewithmarginpars		\mpgmpar@savemarginpars . . . . .	<u>5</u> , 29
minipagewithmarginpars . . . . .	<u>1</u>		
F		T	
\fi . . . . .	16	\temp@a . . . . .	9, 10