Das LaTeX-Paket `abb`

Daniel Baumgartner
Adresse `bg@geometria.ch`

Version 4.0 von 1.6.2017

Das Paket `abb` stellt Funktionen zur Verfügung, die nützlich für das Einbinden von Figuren sind. Des weiteren lassen sich auch Objekte einbinden, die vorgängig mit `\def` (oder `\newcommand`) *ohne* Parameter definiert wurden.

Die Makros sind so konzipiert, dass sie den üblichen Bedarf abdecken.

Mit `\usepackage{abb}` werden die Makros eingelesen. Das Paket ist auf `www.geometria.ch` erhältlich.

Inhaltsverzeichnis

1 Einbinden von Figuren und sonstigem Material

Die Funktion `\Abb` und die Umgebung `AbbTxt` erlauben auf einfache Weise das Einbinden von Graphiken und sonstigem Material wie Tabellen etc. an genau vorgeschriebenen Stellen. Die Graphiken sind entweder externe Dateien oder Material, welches durch ein Makro innerhalb der TeX-Datei definiert wird. Dies können `pstricks`-Figuren, Bilder einer `picture`-Umgebung, oder Tabellen etc. sein. Das Einbinden solchen Materials geschieht in gleicher Weise wie bei externen Dateien. Insbesondere lassen sich derartige Figuren automatisch numerieren und auf einfache Weise mit einer Legende versehen.
Sind zum Beispiel die folgenden Makros

```
\def\figurA{
    \begin{pstricks}(..,..)
       ...
       ...
    \end{pstricks}
}
```

```
\def\figurB{
    \begin{picture}(..,..)
       ...
       ...
    \end{picture}
}
```

```
\def\figurC{
    \begin{tabular}{..}
       ...
       ...
    \end{tabular}
}
```

innerhalb der Hauptdatei definiert, so können sie mit `\Abb{figurA.def}` an eine bestimmte Stelle gesetzt werden.

Eine externe `foo.eps`-Datei wird mit `\Abb{foo.eps}` eingebunden, und genauso eine gewöhnliche `.tex`-Datei. Die Syntax ist für alle Arten von Material ein und dieselbe.

2 Syntax

2.1 Setzen von Bildmaterial

`\Abb` Das Makro `\Abb` setzt eine Figur an eine genau vorgegebene Stelle. Es hat zwei Argumente, wovon eines optional ist. Die Syntax ist

`\Abb[`⟨*Optionen von abb, Optionen von includegraphics*⟩`]{`⟨*Material*⟩`}`

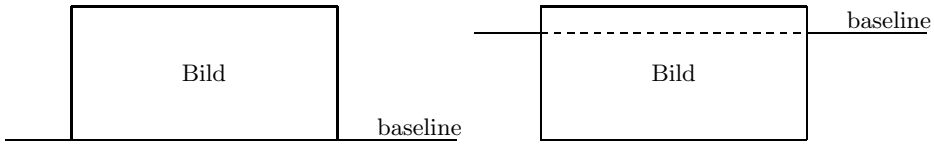
⟨*Material*⟩ ist entweder eine externe Datei oder die Bezeichnung eines Makros. Die Optionen von `includegraphics` sind nur relevant bei `eps`-, `ps`-, `jpg`, `jpeg`-, `pdf`, `png`- Dateien. Diese Optionen werden bei `.tex` und `.pst`-Dateien ignoriert. Beim Suffix `psf` wird zusätzlich die einschlägige `tex`-Datei mit den Informationen für `psfrag` eingelesen.

Mit `\Abb{foo.def}` wird das im Makro `\foo` definierte Material gesetzt. Bei allen anderen Suffixe wird mit `\Abb{foo.xyz}` die Datei `foo.xyz` eingelesen.

Die Reihenfolge der Optionen ist insofern wichtig, als zuerst die Optionen von `abb` und danach die Optionen von `includegraphics` folgen müssen.

Die Optionen von `Abb` sind:

`pos` Mit `pos` wird die Position der Figur in Bezug zur Textzeile gesteuert. `pos=b` zetzt die Figur mit dem unteren Rand auf die Grundzeile. Mit `pos=t` wird die Figur so gesetzt, dass der obere Bildrand den Abstand einer Zeilenhöhe zur Grundzeile aufweist. Dies ist zum Beipsiel nützlich in Tabellen. Die Default-Setzung ist `pos=b`.



`pos=b` (bottom) `pos=t` (top)

`num` Wenn die automatische Figurennumerierung aktiviert ist, so kann mit `num=false` die fortlaufende Numerierung unterdrückt werden. Default ist `num=false`.

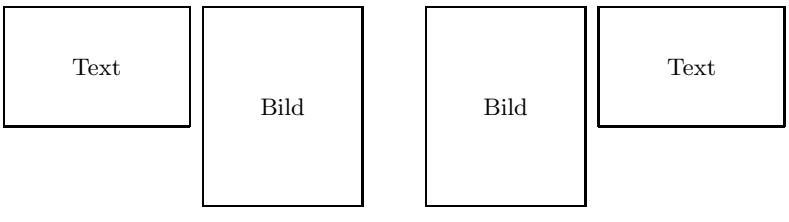
`label` Bei aktiver automatischer Figurennumerierung kann ein Label definiert werden. Die Referierung erfolgt wie gewohnt mit `\ref{..}` oder `\pageref{..}`. Der Verweis wird gemäss der Numerierungsform gestezt, weche in der Definition von `\AbbFigurNum` festgelegt ist.

`AbbTxt` Die Umgebung `AbbTxt` setzt eine Figur neben einen Text, sodass der obere Rand der Figur mit dem oberen Textrand übereinstimmt. Die Syntax ist

`\begin{AbbTxt}[`⟨*Optionen*⟩`]{`⟨*Material*⟩`}`
⟨*Text*⟩
`\end{AbbTxt}`.

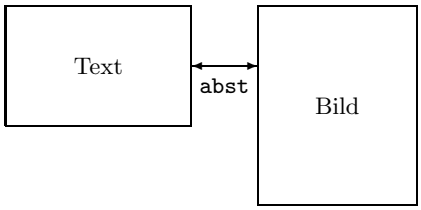
Die Optionen sind dieselben wie bei `\Abb` (wobei `pos` keine Wirkung hat). Zusätzlich stehen noch die folgenden Optionen zur Verfügung.

`ort` Mit `ort` wird die Position des Bildes in Bezug zum Text festgelegt. `ort=r` stezt das Bild links vom Text, `r` rechts davon. Default-Wert ist `ort=r`.



`ort=r` `ort=l`

`abst` Mit `abst` wird er Abstand zwischen dem Text und dem Bild gesteuert. Default-Wert ist `10pt`.



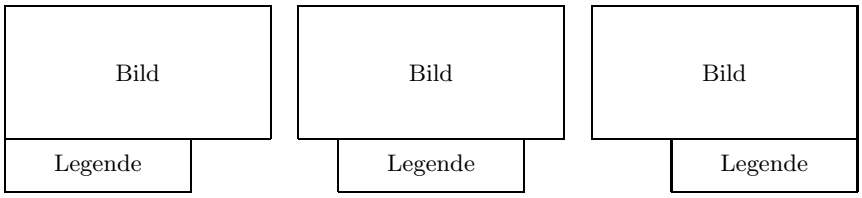
Die Option `pos` ist hier irrelevant und wird ignoriert.

`\AbbLegende` Mit `\AbbLegende` wird ein Legendentext unterhalb der Figur gesetzt. Die Funktion hat zwei Argumente, wobei das erste optional ist. Die Syntax lautet

`\AbbLegende[`⟨*Optionen*⟩`]{`⟨*Legendentext*⟩`}`

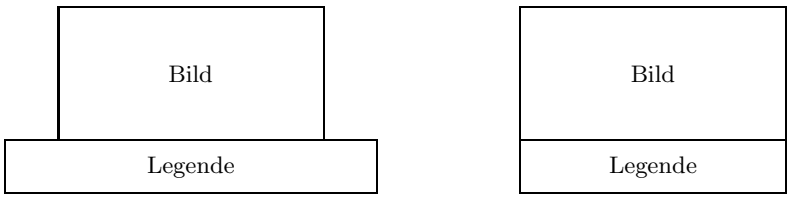
Die zur Verfügung stehenden Optionen sind die folgenden.

`legpos` Mit `legpos=c` wird eine Legende zentriert in Bezug zum Bild gesetzt. Mit links- bzw. rechtsbündig gesetzt. Default ist `legpos=c`.



`legpos=l` `legpos=c` `legpos=r`

`breite` Mit einer Längenangabe `breite=300pt` kann die Breite des Legendentextes gesteuert werden. Default ist das leere Zeichen `breite=`. Der Legendtext ist dann eine einzige Zeile.
Mit `breite=abb` entspricht die Legendenbreite genau der Bildbreite.



`breite=`*dimension* `breite=abb`

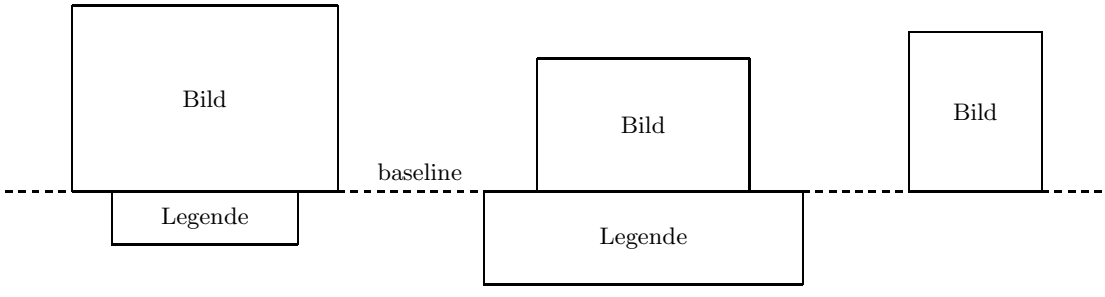
Die Legende ist immer vor der Figur zu definieren.

```
\AbbLegende{..}
\Abb[..]{..}
```

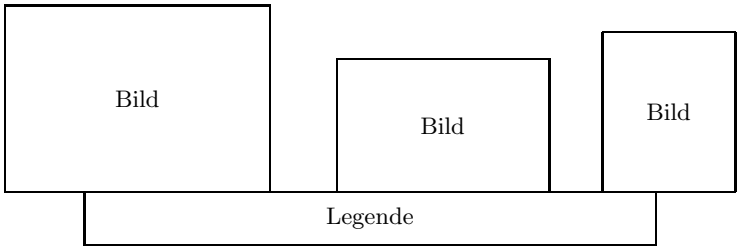
```
\AbbLegende{..}
\begin{AbbTxt}[..]{..}
...
\end{AbbTxt}
```

```
\AbbLegende{..}
\begin{MultiAbb}[..]{..}
...
\end{MultiAbb}
```

Mehrere Figuren auf einer Zeile werden so gesetzt, dass ihr unterer Rand auf die Grundlinie zu liegen kommt, und ihr Legendext direkt unterhalb.



`MultiAbb` Diese Umgebung ist in erster Linie dazu gedacht, Figuren auf einer Zeile mit einem gemeinsamen Legendentext zu versehen. Falls die automatische Numerierung aktiviert ist, wird sie bei den einzelnen Figuren unterdrückt. Die Figuren erhalten dann im gemeinsamen Legendentext eine gemeinsame Figurnummer.



Die Syntax lautet

`\begin{MultiAbb}[`⟨*Optionen*⟩`]{`⟨*Material*⟩`}\end{MultiAbb}`

Die Umgebung erzeugt eine `vbox` mit dem ⟨*Material*⟩. Eine allfällig vorhandene Legende folgt als `vbox` auf einer neuen Zeile.

Die Optionen von `MultiAbb` sind `label` und `pos`, sowie

`abblaenge` Mit einer Längenangabe `abblaenge=300pt` oder `abblaenge=0.7\hsize` kann die Breite der Umgebung festgelegt werden. Default ist `\hsize` (die Textbreite).

`\abbset` Mit `\abbset` werden Optionen global gesetzt. Die Defaultwerte sind

```
pos=b,ort=r,num=false,label=,abst=10pt,breite=,legpos=c,abblaenge=\hsize
```

Zum Beispiel werden mit `\abbset{num=false,legpos=l}` die Figuren automatisch numeriert und der Legendentext standardmässig linksbündig gesetzt.

`\AbbPfad` Sind die Figuren in einem Unterverzeichnis `figuren/` abgelegt, so kann mit der Angabe `\AbbPfad{figuren/}` der Pfad gesetzt werden. So kann anstelle von `\Abb{figuren/foo.tex}` das Material mit `\Abb{foo.tex}` eingelesen werden.

2.2 Automatische Figurennumerierung

Mit `\abbset{num=true}` wird die automatische Figurennumerierung aktiviert. Sie kann temporär bei einer Figur mit der Option `num=false` ausgeschaltet werden.

Um die Gestaltung der Numerierung und Namensgebung möglichst flexibel zu halten, stehen die Funktionen `\AbbFigurNum` und `\AbbNumForm` zur Verfügung.

`\AbbFigurNum`

Diese Funktion legt Darstellung der Figurnummer fest. Im Paket `abb` ist die Funktion gemäss nebenstehendem Code vordefiniert.
In der Variablen `\figurNr` ist die aktuelle Figurnummer gespeichert.

```
\def\AbbFigurNum{%
\number\figurNr%
}%
```

Mit `\renewcommand{\AbbFigurNum}` oder `\def\AbbFigurNum` kann die Gestaltung angepasst werden. Mit

```
\renewcommand{\AbbFigurNum}{%
    \thesection.\number\figurNr%
}%
```

wird vor der Figurnummer noch die Sektionsnummer vorangestellt und mit einem Punkt getrennt.

Die in `\AbbFigurNum` definierte Numerierungsform wird auch von `\char92ref{..}` verwendet, wenn ein Label gesetzt wird.

\AbbNumForm

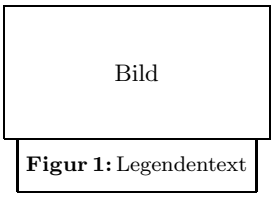
Mit \AbbNumForm kann die Art und Weise der Namensgebung frei gestaltet werden. Im Paket abb ist die Funktion gemäss nebenstehendem Code vordefiniert. Sie hat ein Argument, die Länge des Legendentexts. Mit der bedingten Verzweigung \ifdim#1=0pt kann so zum Beispiel das Trennzeichen weggelassen werden, wenn kein Legendentext vorhanden ist.

```
\def\AbbNumForm#1{%
    Figur \AbbFigurNum%
    \ifdim#1=0pt\else:\,\fi
}%
```

Mit \renewcommand{\AbbNumForm}{#1} oder \def\AbbNumForm#1 kann die Gestaltung angepasst werden.

Beispiel 1. Die Abbildungen werden fettgedruckt **Figur** genannt. Der Legendentext wird durch das Trennzeichen ":" separiert.

```
\def\AbbNumForm#1{%
    {\bf Figur\,\AbbFigurNum}%
    \ifdim#1=0pt\else:\,\fi
}%
```



Figur 1: Legendentext

Beispiel 2. Eine Variante mit dem Namen Fig. und dem Trennzeichen ":". Das Trennzeichen entfällt, wenn kein Legendentext vorhanden ist.

```
\def\AbbNumForm#1{%
    Fig.\,\AbbFigurNum%
    \ifdim#1=0pt\else:\,\fi
}%
```

Fig. 2: Etwas Text

Beispiel 3. Die Abbildungen werden fettgedruckt **Abbildung** genannt. Nach der Nummer wird ein Punkt gesetzt. Ein Trennzeichen entfällt. Des weiteren soll bei mehrzeiligem Legendentext ab der zweiten Zeile um 10pt eingerückt werden.

```
\def\AbbNumForm#1{%
     \hangindent10pt\hangafter1
     {\bf Abbildung\,\AbbFigurNum.}\,%
}%
```

Abbildung 3. Zwei, drei Worte als Textbeispiel.

2.3 Fussnoten

\AbbFootnote

Diese Funktion ermöglicht das Setzten von Fussnoten innerhalb der AbbTxt-Umgebung und im Legendentext. Die Syntax lautet

\AbbFootnote{⟨*Text*⟩}

2.4 Dateiformate

In der folgenden Tabelle sind die Suffixe aufgelistet, welche vom Paket `abb` erkannt werden. Mit Ausnahme von `.def` handelt es sich um externe Dateien.

Allgemein	
`.tex`	TEX-Datei
`.def`	Makro
TEX und DVI	
`.tex` oder `.pst`	`pstricks`-Dateien
`.eps`	eps-Figuren
`.psf`	eps-Figuren mit `psfrag`-Datei
`.ps`	POSTSCRIPT-Datei
pdf-LATEX	
`.jpg` oder `.jpeg`	JPG-Dateien
`.pdf`	PDF-Datei
`.png`	png-Datei

Bemerkungen. `eps`-Figuren lassen sich skalieren, wobei aber Beschriftungen gleichzeitig mitskaliert werden, was eine zu kleine oder zu grosse Schrift als unerwünschten Nebeneffekt zur Folge hat. Einen Ausweg bietet das Paket `psfrag`. Diese Paket erlaubt Schriften in `eps`-Dateien durch TEX-Ausdrücke zu substituieren. Das Paket `abb` bietet für diesen Fall eien einfache Handhabung. Bei gegebener `foo.eps`-Datei erzeuge man eine gleichnamige Datei `foo.pst` welche die `psfrag`-Information der Substitutionen enthält. Mit `\Abb{foo.psf}` oder mit der Umgebung `AbbTxt` wird automatisch die `eps`-Datei zusammen mit der `psfrag`-Datei eingelesen.

PSTricks-Figuren können nur bedingt skaliert werden. Dies wegen eines Bugs im Paket `pstricks`: Kreise und Bögen werden nicht mitskaliert, währenddessen alle anderen Objekte skaliert werden. PSTricks-Figuren müssen deshalb immer in Originalgrösse erstellt werden.

3 TEX-Code

```
\ProvidesPackage{abb}[2017/06/01 v4.0]
\catcode`\@=11
```

`\Abb` Das Makro `\Abb` hat zwei Argumente, wovon eines optional ist. Die eigentliche Implementierung wird in `\Abb@[`⟨*Optionen*⟩`]{`⟨*Material*⟩`}`vorgenommen.
Zunächst sind die Optionen zu behandeln. Die tut die Funktion `\filter@opt@`, welche auch von `\AbbTxt` und `\AbbLegende` verwendet wird. Daher wird der Funktionsname temporär der Variablen `\abb@alg` zugewiesen.
Falls kein optionales Argument vorhanden ist, wird direkt `\Abb@` aufgerufen. Ansonsten kommt `\filter@opt@` zum Zug.

```
\def\Abb{%
    \let\abb@alg\Abb@%
    \@ifnextchar[{\filter@opt@}{\Abb@[]}%
}%
```

`\AbbTxt` Das Makro `\AbbTxt` ist Teil der gleichnamigen Umgebung und ist zuständig für das Setzen einer Figur neben einem Text. Es hat zwei Argumente, wovon eines optional ist. Die eigentliche Implementierung wird in `\AbbTxt@[`⟨*Optionen*⟩`]{`⟨*Material*⟩`}` vorgenommen. Zunächst sind die Optionen zu behandeln. Wie bei `\Abb` wird der Funktionsname temporär der Variablen `\abb@alg` zugewiesen.
Falls kein optionales Argument vorhanden ist, wird direkt `\AbbTxt@[]` aufgerufen. Ansonsten kommt `\filter@opt@` zum Zug.

```
\def\AbbTxt{%
    \let\abb@alg\AbbTxt@%
    \@ifnextchar[{\filter@opt@}{\AbbTxt@[]}%
```

```
}%
```

`\filter@opt@` Das Makro `\filter@opt@` fängt den Fall des leeren optionalen Arguments ab. Falls `#1` leer ist, so wird `\abb@alg` aufgerufen, ansonsten werden mit `\filter@opts` die Optionen gestezt.

```
\def\filter@opt@[#1]{%
    \def\@arg{#1}
    \ifx\@arg\empty%
        \let\next\abb@alg%
    \else%
        \let\next\filter@opts%
    \fi%
    \next[#1]
}%
```

`\filter@opts` Der Liste von Optionen wird das Token `\relax` als Markierung des Listenendes hinzugefügt. Die eigentliche Setztung der Werte der optionalen Parameter übernimmt `\flt@opt`.

```
\def\filter@opts[#1=#2]{%
    \flt@opt[#1=#2,\relax]
}%
```

`\flt@opt` Die Variablen, welche die Werte der Optionen enthalten, werden mit ⟨*name*⟩`@opt` bezeichnet, wobei ⟨*name*⟩ den Optionennamen bezeichnet, zum Beipsiel `pos@opts` oder `label@opt`.
Die Liste der Optionen wird rekursiv abgearbeitet. Die erste Option ist von der Form `#1=#2`, gefolgt vom Rest `#3`. Wenn `#1@opt` unbekannt ist, so wird `#1=#2,#3` als Optionen von `\includegraphics` interpretiert. In diesem Fall kommt `\flt@opt@i` zum Zug.
Andernfalls wird der Variablen `#1@opt` der Wert `#2` zugewiesen. Sodann wird geprüft, ob `#3` das Listenende (also das Token `\relax`) ist. Trifft dies zu, so wird die Rekursion mit `\flt@opt@ii` beendet. Andernfalls wird `\next` die Funktion `\flt@opt@iii` zugewiesen, welche dann (rekursiv) `\flt@opt` aufruft.

```
\def\flt@opt[#1=#2,#3]{%
    \@ifundefined{#1@opt}{\let\next\flt@opt@i}{%
        \expandafter\xdef\csname #1@opt\endcsname{#2}%
        \ifx#3\relax%
            \let\next\flt@opt@ii%
        \else%
            \let\next\flt@opt@iii%
        \fi%
        }%
    \next[#1=#2,#3]%
}%
```

`\flt@opt@i` `\flt@opt@ii` `\flt@opt@iii` Die drei Fälle für die Rekursion in `\flt@opts`. Bei `\flt@opt@i` enthält `#1` die Optionen von `\includegraphics`. Bei `\flt@opt@ii` ist `#1` leer. In diesen beiden Fällen wird die Rekursion in `\flt@opts` abgebrochen und mit `\abb@alg` (welche `\Abb@`, `\AbbTxt@` oder `\AbbLegende@` ist) weitergefahren. Die Funktion `\flt@opt@iii` führt `\flt@opt` mit dem Rest der `#3` der Liste aus.

```
\def\flt@opt@i[#1,\relax]{%
    \abb@alg[#1]%
}%
\def\flt@opt@ii[#1,\relax]{%
    \abb@alg[]%
}%
\def\flt@opt@iii[#1=#2,#3]{%
    \flt@opt[#3]%
}%
```

`\bild@box` In dieser Box wird das Bildmaterial gesetzt, inklusive allfällig vorhandener Legende.

```
\newbox\bild@box%
```

\Abb@ Die Funktion \Abb@ ruft \erzeuge@abb@box auf, welche doe Bildbox \bild@box erzeugt und ausgibt.

```
\long\def\Abb@[#1]#2{%
    \erzeuge@abb@box[#1]{#2}%
    \box\bild@box%
}%
```

\text@box \text@breite \temp@linewidth \@flagge Die Umgebung AbbTxt benötigt einige lokale Variablen. Die Boxvariable \text@box enthält den Text neben der Figur. Die Dimensionsvariable \text@breite ist die Textbreite neben der Figur. Die Länge \temp@linewidth speichert temporär den Wert von \linewidth, damit AbbTxt auch in den LaTeX-Umgebungen wie itemize etc. anwendbar ist. Schliesslich steuert die numerische Variable \@flagge ob die Figur rechts (0) oder links (1) des Textes gesetzt wird.

```
\newbox\text@box%
\newdimen\text@breite
\newdimen\temp@linewidth
\newcount\@flagge%
```

\AbbTxt@ Es folgt die Implementation der Umgebung AbbTxt. Das erste Argument #1 ist leer oder enthält die Optionen für \includegraphics. Das zweite Argument enthält die Information über das als Abbildung zu setzende Material.
Wenn die Variable \@flagge den Wert 0 hat, so wird die Abbildung rechts vom Text gesetzt. Im Falle ort=l wird der Variablen @\flagge der Wert 1 zugewiesen.
Sodann wird \erzeuge@abb@box aufgerufen.

```
\long\def\AbbTxt@[#1]#2{%
    \global\@flagge=0%
    \if\ort@opt l%
        \global\@flagge=1%
    \fi%
    \erzeuge@abb@box[#1]{#2}%
```

In der Box \bild@box ist jetzt die Figur enthalten. Nun wird die Textbreite \text@breite aufgrund der Bildbreite und des Abstandes zwischen dem Text und dem Bild berechnet. Letzterer ist in \abst@opt gespeichert.

```
    \text@breite=\linewidth%
    \advance\text@breite by-\wd\bild@box%
    \advance\text@breite by-\abst@opt%
```

Die Information in der Längenvariable \linewidth wird in \temp@linewidth temporär gespeichert und ihr sodann die Länge \text@breite zugeordnet. Dies ist nötig, damit sich die Umgebung AbbTxt auch in den Listenumgebungen von LaTeX korrekt verhält. Bei verschachtelten Listen ist der linke Randabstand noch anzupassen (\@listdepth und \@totalleftmargin sind LaTeX-Variablen für das Listenkonstrukt).

```
    \let\temp@linewidth=\linewidth%
    \let\linewidth=\text@breite%
    \ifnum \@listdepth >0%
        \advance\@totalleftmargin-\leftmargin%
    \fi%
```

Nun wird die vertikale Box \text@box mit dem neben das Bild zu setzendem Text erzeugt. Mit \bgroup wird die Box geöfnet...

```
    \setbox\text@box=\vbox\bgroup\hsize=\text@breite%
}%
```

\endAbbTxt ... und mit \egroup am Ende der Umgebung AbbTxt wieder geschlossen. Der Längendimension \linewidth wird nun wieder ihr ursprünglicher Wert zugewiesen.

```
\def\endAbbTxt{%
    \egroup%
    \linewidth=\temp@linewidth%
```

Die Boxen werden so angeordnet, dass links die Box `1` und rechts die Box `0` zu stehen kommt. In der Box mit der Nummer `\@flagge` wird das Bild abgelegt (`\box0` bei rechtsseitigem Bild, `\box1` bei linksseitigem Bild). Ist a die Boxnummer des Bildes, so ist $-a+1$ die Boxnummer des Textes.

```
    \bgroup
    \setbox\@flagge=\box\bild@box%
    \multiply\@flagge by-1%
    \advance\@flagge by1%
    \setbox\@flagge=\vbox{\unvbox\text@box}%
```

Die ganze Box wird nun auf eine neue Zeile gesetzt. Der Buchstabe `I` dient als Information zur Bestimmung einer gewöhnlichen Textzeilenhöhe.
Nun werden die Text- und Bild-Boxen so nebeneinander gesesetzt, dass sie oben bündig sind. Weil `\hrule` eine Zeile beansprucht, ist die ganze hbox noch um die Zeilenhöhe `\ht3` nach oben zu verschieben. Schliesslich soll am Ende noch ein Leerraum von `2\lineskip` eingefügt werden.

```
    \par%
    \setbox3=\hbox{I}%
    \rightline{%
    \hbox to\linewidth{%
        \raise\ht3\vtop{\hrule height 0pt\box1
        \hrule height0pt}%*
        \hfill%
        \raise\ht3\vtop{\hrule height 0pt\box0}%
        }%
    }%
    \vskip2\lineskip%
    \egroup
```

Schliesslich werden noch die Optionsvariablen `abst`, `ort` auf ihren ursprünglichen Wert zurückgesetzt, und allfällig vorhandene Fussnoten gesetzt.

```
    \ruecksetz@opts{ort,abst}%
    \Abbfootnote@%
}%
```

`\erzeuge@abb@box` Diese Funktion erzeugt die Bildbox inklusive einer Legende. Das erste Argument ist leer oder enthält dieOptionen von `\includegraphics`. Das zweite Argument `#2` ist das Bildmaterial. Zunächst wird das Bildmaterial eingelesen und in der Box `\bild@box` abgelegt.

```
\def\erzeuge@abb@box[#1]#2{%
    \expandafter\@einlesen[#1]#2|%
```

Die Bildbox soll keine Tiefen-, sondern nur Höhen-Dimension haben, damit ihr unterer Rand stets auf der Grundlinie einer Zeile zu liegen kommt. Ihre Tiefe wird auf `0pt` gesetzt, und ihre Höhe ist die Summe ihrer ursprünglichen Höhe und Tiefe.

```
    \begingroup%
    \dimen1=\ht\bild@box%
    \advance\dimen1by\dp\bild@box%*
    \dp\bild@box=0pt%*
    \ht\bild@box=\dimen1%
```

Zunächst wird geprüft, ob eine automatische Nummerierung der Abbildungen aktiviert ist. Trifft dies zu, so ist sicher eine Legende vorhanden, und daher `\@abbleg@` auf `true` zu setzen.

```
    \csname @AbbNum@\num@opt\endcsname%
    \if@AbbNum@\@abbleg@true\fi%
```

Es folgt die Setzung der Bildlegende, welche bereits von `\AbbLegende` erzeugt und in der Box `\leg@box` abgelegt wurde. `\if@abbleg@` ist genau dann `true`, wenn eine Legende vorhanden ist (oder eine automatische Nummerierung erfolgt).
Das Längenregister `\dimen2` misst die Gesamthöhe der Legendenbox.

```
    \dimen2=0pt%
    \if@abbleg@%
        \erzeuge@leg@box%
```

```
        \dimen2=\ht\leg@box%
        \advance\dimen2 by\dp\leg@box%
    \fi%
```

Das Längenregister `\dimen3` misst die Gesamtbreite der Bildbox als Maximum der Bild- und Lgendenboxbreite.

```
    \ifdim\wd\leg@box>\wd\bild@box%
        \dimen3=\wd\leg@box%
    \else%
        \dimen3=\wd\bild@box%
    \fi%
```

Nun wird die Abbildungsbox als vertikale Box erzeugt. Bei einer leeeren Legendenbox wird eine leere Box angefügt.
Je nach Wert der Option `legpos` wird der Legendentext entsprechend gesetzt.

```
    \global\setbox\bild@box=\vbox{%
        \hsize=\dimen3%
        \hbox to\hsize{%
            \if\legpos@opt l\else\hfill\fi
            \box\bild@box
            \if\legpos@opt r\else\hfill\fi%
            }%
        \par%
        \vskip\lineskip%
        \hbox to \hsize{%
            \if\legpos@opt l\else\hfill\fi
            \box\leg@box%
            \if\legpos@opt r\else\hfill\fi%
            }%
        }%
```

Nun werden die Masse der Bildbox definiert. Die Höhe der Box soll die Bildhöhe `\dimen1`, und die Tiefe die Legendhöhe `\dimen2` sein. So ist sichergestellt, dass bei nebeneinandergesetzten Figuren die Legenden auf gleicher Höhe zu liegen kommen.
Eine Ausnahme liegt vor, wenn `\pos@opt` das Zeichen `t` ist. Dann soll die Höhe der Bildbox gleich der Zeilenhöhe `\ht0` sein, und die Tiefe die Differenz von der Gesamthöhe `\dimen2` und `\ht0`.
Schliesslich werden die Höhe und Tiefe von `\bild@box` festgelegt gemäss den aktuellen Werte von `\dimen1` und `\dimen2`.

```
    \if\pos@opt t%
        \setbox0=\hbox{I}%
        \advance\dimen2 by\dimen1%
        \advance\dimen2 by-\ht0%
        \dimen1=\ht0%
    \fi%
    \global\ht\bild@box=\dimen1%
    \global\dp\bild@box=\dimen2%
    \endgroup%
```

Schliesslich wird die Boolsche Variable `\@abbleg@` auf `false` gesetzt, falls keine automatische Nummerierung aktiviert ist, sowie die die Legendenbox als leere Box gespeichert.
Am Ende werden die Werte der Optionen (mit Ausnahme von `abst` wieder auf ihre ursprünglichen Werte gesetzt.

```
    \if@AbbNum@\else\@abbleg@false\fi%
    \xdef\leg@txt{\empty}%
    \ruecksetz@opts{pos,ort,num,label,breite,legpos}%
}%
```

`\@einlesen` Diese Funktion liest die Information über das Bildmaterial ein. Entweder als Datei oder via Makros.

```
\def\@einlesen[#1]#2.#3|{%
    \setbox\bild@box=\vbox{%
```

```
        \hbox{\csname @#3@\endcsname[#1]{#2}}%
        }%
}%
```

`\@pfad` `\AbbPfad` Mit `\AbbPfad` kann ein Pfad zu einem Verzeichnis angegeben werden, in Welchem sich die Figuren befinden. Dieser wird im Token `\@pfad` gespeichert. Default ist der leere Pfad.

```
\newtoks\@pfad
\@pfad={}%
\def\AbbPfad#1{%
    \@pfad={#1}
}%
```

`\@tex@` `\@def@` `\@eps@` `\@psf@` Je nach Suffix sind die folgenden Schritte auszuführen.

```
\def\@tex@[#1]#2{\input{\the\@pfad#2.tex}}%
\def\@def@[#1]#2{\csname#2\endcsname#1}%
\def\@pst@[#1]#2{\input{\the\@pfad#2.pst}}%
\def\@eps@[#1]#2{\includegraphics[#1]{\the\@pfad#2.eps}}%
\def\@psf@[#1]#2{\input{\the\@pfad#2.psf}\includegraphics[#1]{\the\@pfad#2.eps}}%
\def\@ps@[#1]#2{\includegraphics[#1]{\the\@pfad#2.ps}}%
\def\@jpg@[#1]#2{\includegraphics[#1]{\the\@pfad#2.jpg}}%
\def\@jpeg@[#1]#2{\includegraphics[#1]{\the\@pfad#2.jpeg}}%
\def\@png@[#1]#2{\includegraphics[#1]{\the\@pfad#2.png}}%
\def\@pdf@[#1]#2{\includegraphics[#1]{\the\@pfad#2.pdf}}%
```

`\leg@box` `\if@abbleg@` Für die Erzeugung der Legendenbox sind einige Variablen nötig. In der `\leg@box` wird der Legendtext gespeichert. Sie soll vorgängig leer sein. Die Boolsche Variable `\if@abbleg@` ist genau dann `true`, wenn eine Legende vorhanden ist. Ihr Default-Wert ist `false`.

```
\newbox\leg@box%
\setbox\leg@box=\hbox{\empty}
\newif\if@abbleg@
\@abbleg@false%
```

`\if@AbbNum@` `\figurNr` Die Boolsche Variable `\if@AbbNum@` ist genau dann `true`, wenn die Figuren automatisch numeriert werden sollen. Ihr Default-Wert ist `false`. Im (globalen) Zähler `\figurNr` ist die aktuelle Figurnummer gespeichert. Er ist initial auf `0` gesetzt.

```
\newif\if@AbbNum@
\newcount\figurNr
\figurNr=0
```

`\AbbFigurNum` `\AbbNumForm` Die Konstante `AbbFigurNum` enthält die Gestaltung der automatischen Numerierung.
Die Funktion `\AbbNumForm` setzt die automatische Numerierung. Das Argument `#1` enthält die Information über die Breite des Legendentexts. Dies erlaubt den Fall zu steuern, wenn zwischen Figurname und Legendentext ein Trennzeichen gesetzt werden soll, welches im Falle einer leeren Legende wegfallen soll.

```
\def\AbbFigurNum{%
    \number\figurNr%
}%
\def\AbbNumForm#1{%
    Figur \AbbFigurNum%
    \ifdim#1=0pt\else:\,\fi
}%
```

`\setze@label` Diese Funktion definiert ein Label. Wenn `\label@opt` nicht leer ist, so wird die Information der Figurennummer und der Seitennummer via `\newlabel` in die `.aux`-Datei geschrieben.

```
\def\setze@label{
    \ifx\label@opt\empty
    \else
        \immediate\write\@mainaux{\string\newlabel{\label@opt}{{\AbbFigurNum}{\thepage}}}%
    \fi%
    \gdef\label@opt{}%
}
```

\leg@breite
\leg@txt

Für die Erzeugung einer Legendenbox benötigt man die Dimensionsvariable \leg@breite, welche die Länge des Legendentextes enthält, und die Variable \leg@txt welche den Legendentext beinhaltet. Sie wird vorgängig leer gesetzt.

```
\newdimen\leg@breite
\def\leg@txt{\empty}%
```

\AbbLegende

Zunächst wird die Boolsche Variable \@abbleg@ auf true gesetzt. Damit kann jederzeit abgefragt werden, ob eine Legende vorhanden ist oder nicht. Sodann ist das optionale Argument zu untersuchen, bevor der eigentliche Algorithmus \AbbLegende@[#1]#2 aufgerufen wird. Hierbei wird auf den allgemeinen, oben beschriebenen Algorithmus \filter@opt@ zurückgegriffen. Dort werden alle Optionen gesetzt.

```
\def\AbbLegende{%
    \@abbleg@true%
    \let\abb@alg\AbbLegende@%
    \@ifnextchar[{\filter@opt@}{\AbbLegende@[]}%
}
```

\AbbLegende@

Die Funktion \AbbLegende@ speichert den Legendentext in der Variablen \leg@txt. Das Argument [#1] ist nur der syntaktischen Korrektheit halber notwendig (damit \filter@opt@ auch auf diesen Fall angewandt werden kann).

```
\def\AbbLegende@[#1]#2{%
    \gdef\leg@txt{#2}%
}
```

\erzeuge@leg@box

Diese Funktion wird von \erzeuge@abb@box aufgerufen, nachdem die Information über die Bilddimensionen vorhanden sind, und \if@abb@leg@ den Boolschen Wert true hat.
In der box0 wird der Legendentext abgelegt. Die zunächst leere box1 wird später eine allenfalls vorhandene Figurnummerierung enthalten. Falls eine automatische Figurennummerierung aktiviert ist (\if@AbbNum@ hat dann den Wert true) wird die Figurennummer \figurNr um 1 erhöht. Sodann wird die Gestaltung der Figurennummerierung mit \AbbNumForm vorgenommen und das ganze Material in der box1 abgelegt. Schliesslich wird mit \setze@label ein allfällig vorhandenes Label gesetzt.

```
\def\erzeuge@leg@box{%
    \begingroup%
    \setbox0=\hbox{\leg@txt}%
    \setbox1=\hbox{}%
    \if@AbbNum@%
        \global\advance\figurNr by 1%
        \setbox1=\hbox{%
            \AbbNumForm{\wd0}}%
        \setze@label%
    \fi%
```

dimen1 ist die Summe der Boxlängen der beiden Boxen box0 und box1, also die Gesamtlänge der Legende inkl. der Figurennummerierung. Die box3 enthält die Information des Buchstabens I zur späteren Bestimmung der Zeilenhöhe.

```
    \dimen0=\wd0%
    \advance\dimen0by\wd1%
    \setbox3=\hbox{I}%
```

Es folgt die Bestimmung der Breite \leg@breite des Legendentextes unterBerücksichtigung des Werts in der Optionsvariable \breite@opt. Zunächst wird der Variable \leg@breite per Default die Breite der Figurbox zugeordnet. Wenn \breite@opt leer ist, so erhält \leg@breite den Wert \dimen0, d.h. der Legendentext ist dann einzeilig mit seiner Breite \dimen0. Sonst wird zunächst geprüft, ob \breite@opt mit der Zeichenkette abb identisch ist. Falls dies zutrifft bleibt \leg@breite unverändert die Bildbreite. Ansosten muss \breite@opt eine Länegnangabe sein. Falls die Breite \dimen0 kleiner als die Legendenbreite ist, so wird \leg@breite die Länge dimen0 zugewiesen, andernfalls die Breiteangabe von \breite@opt.

```
    \leg@breite=\wd\bild@box%
```

```
    \ifx\breite@opt\empty%
        \leg@breite=\dimen0%
    \else%
        \def\@arg{abb}%
        \ifx\@arg\breite@opt%%
        \else%
            \ifdim\dimen0<\breite@opt%
                \leg@breite=\dimen0%
            \else%
                \leg@breite=\breite@opt%
            \fi%
        \fi%
    \fi%
```

Nun wird die Legendenbox als `vbox` erzeugt, deren `hsize` der Legendenbreite `\leg@breite` entspricht. Zunächst wird ein fixer Zeilenabstand zum unteren Rand der Bildbox gesetzt (das 6-fache von `\lineskip`). Sodann werden die Boxen `box1` und `box0` nebeneinander gesetzt und die äussere `hbox` entfernt. Wenn die Legendenbreite grösser als die Bildbreite ist, wird nichts weiter getan. Ansonsten wird noch eine `hbox` mit der Breite der Legendbox gesetzt. Schliesslich werden noch allfällig vorhandene Fussnoten behandelt.

```
    \global\setbox\leg@box=\vbox{\hsize=\leg@breite%
        \vskip6\lineskip%
        \ifdim\dimen0>\wd\bild@box%
            \if@AbbNum@\AbbNumForm{\wd0}\fi\leg@txt%
        \else%
            \hbox to\hsize{%
            \if@AbbNum@\AbbNumForm{\wd0}\fi\leg@txt}%
        \fi%
        \vskip\ht3}%
    \endgroup%
    \Abbfootnote@%
}%
```

`\setz@lok@opt` In den nächsten Makros folgt das Speichern und Setzen von Optionen. Die Variablen sind von der Form `gb@`⟨*optnam*⟩`@opt`, `lk@`⟨*optnam*⟩`@opt` oder ⟨*optnam*⟩`@opt`. Hierbei ist ⟨*optnam*⟩ einer der gültigen Optionsvariablen. In `gb@`⟨*optnam*⟩`@opt` ist der globale Wert gespeichert. In ⟨*optnam*⟩`@opt` sind vor jedem Makroaufruf die globalen Werte gespeichert. Erst beim Aufruf eines Makros wird ein temoprärer Wert zugewiesen, falls dort Optionen verwendet werden. Analog verhält sich `lk@`⟨*optnam*⟩`@opt`. Sie wird in der lokalen Umgebung `MultiAbb` gebraucht.
Das Makro `\setz@lok@opt` definiert die Werte für ⟨*optnam*⟩`@opt` oder `lk@`⟨*optnam*⟩`@opt`. Das erste Argument `#1` ist eine Variable der Form ⟨*optnam*⟩`@opt` oder `lk@`⟨*optnam*⟩`@opt`. Das zweite Argument `#2` ist eine Variable der Form `@gb`⟨*optnam*⟩`@opt` oder `lk@`⟨*optnam*⟩`@opt`. Es handelt sich um die Zuweisung `\def#1{#2}`.

```
\def\setz@lok@opt#1#2{%
    \expandafter\xdef\csname #1\endcsname{\csname#2\endcsname}%
}
```

`\opts@lokal` `\@optii` Die Funktion `\opts@lokal` ruft rekursiv `\setz@lok@opt` aufgrund der Informationen in der Liste `[#1,#2]` von Opionsvariablen auf. Das Argument `#3` ist `lk@` oder leer. Das Argument `#4` ist `lk@` oder `gb@`. Falls `#2` identisch mit `\relax` ist, ist das Listenende erreicht, und die Rekursion wird mit `\@optii` abgebrochen.

```
\def\opts@lokal[#1,#2]#3#4{%
    \setz@lok@opt{#3#1@opt}{#4#1@opt}%
    \ifx#2\relax%
        \let\next\@optii%
    \else%
        \let\next\opts@lokal%
    \fi%
    \next[#2]{#3}{#4}%
```

```
}%
\def\@optii[#1]#2#3{}
```

`\abbset` Dieses Makros erlaubt die globale Setzung der optionalen Variablen. Das Argument ist eine Liste der Form ⟨*Optionsvariable=Wert,..*⟩. Im Falle der leeren Liste soll mit `\abbset@@i` nichts getan werden. Andernfalls wird der Liste `#1` das Token `\relax` als Endmarkierung hinzugefügt und mit `\abbset@@` weitergefahren.

```
\def\abbset#1{%
    \def\@arg{#1}%
    \ifx\@arg\empty%
        \let\next\abbset@@i%
    \else%
        \let\next\abbset@@%
    \fi%
    \next[#1,\relax]%
}%
```

`\abbset@@` `\abbset@@i` Das Makro `\abbset@@` arbeitet die Liste mit den Zuweisungen solange rekursiv ab, bis die Endmarkierung `\relax` erreicht ist. Allen drei Variablen `gb@`⟨*optnam*⟩`@opt`, `lk@`⟨*optnam*⟩`@opt`, ⟨*optnam*⟩`@opt` wird der globale Wert zugewiesen. Am Ende der Liste wird mit `\abbset@@i` die Rekursion abgebrochen.

```
\def\abbset@@[#1=#2,#3]{
    \expandafter\xdef\csname gb@#1@opt\endcsname{#2}%
    \expandafter\xdef\csname lk@#1@opt\endcsname{#2}%
    \expandafter\xdef\csname #1@opt\endcsname{#2}%
    \ifx#3\relax\else\abbset@@[#3]\fi%
}%
\def\abbset@@i[#1]{}
```

`\abbset` Hier werden die Grundeinstellungen vorgenommen.

```
\abbset{pos=b,ort=r,num=false,label=,abst=10pt,breite=,legpos=c,abblaenge=\hsize}
```

`\setz@opts` Die Variablen `lk@`⟨*optnam*⟩`@opt`, ⟨*optnam*⟩`@opt` erhalten mit `\setz@opts` die globalen Werte der in der Liste `#1` enthaltenen Optionsvariablen zugewiesen. Beim Aufruf von `\opts@lokal`, welche diese Zuweisung vornimmt, wird am Ende der Liste das Token `\relax` hinzugefügt. Es hat die Bedeutung einer Markierung des Listenendes.

```
\def\setz@opts#1{%
    \opts@lokal[#1,\relax]{}{gb@}%
    \opts@lokal[#1,\relax]{lk@}{gb@}%
}%
```

`\ruecksetz@opts` Am Ende der Funktion `\erzeuge@abb@box` werden alle im Argument `#1` aufgeführten Optionen auf ihre globalen Werte gesetzt.

```
\def\ruecksetz@opts#1{%
    \opts@lokal[#1,\relax]{}{gb@}%
    \opts@lokal[#1,\relax]{}{lk@}%
}%
```

`\multileg@box` Für `MultiAbb` wird die Box `\multileg@box` verwendet. Sie enthält das Legendenmaterial.

```
\newbox\multileg@box
```

`\MultiAbb` Die Funktion verzweigt nach `\MultiAbb@`. Wenn keine optionalen Argumente vorhanden sind, wird sie mit dem leeren optionalen Argument aufgerufen.

```
\def\MultiAbb{%
    \@ifnextchar[{\MultiAbb@}{\MultiAbb@[]}%
}
```

`\MultiAbb@` Zunächst werden die Optionen gesetzt, welche lokal in der Umgebung `MultiAbb` ihre Gültigkeit haben. Sodann wird `@AbbNum@` je nach Belegung von `\num@opt` auf `true` oder `false` gesetzt. Damit hat man die Information ob eine automatische Numerierung aktiviert ist.

```
\long\def\MultiAbb@[#1]{%
    \opts@multiabb[#1]%
    \csname @AbbNum@\num@opt\endcsname%
```

Die folgenden Zeilen behandeln den Fall, wenn ein Legendentext (über mehrere Figuren hinweg) gesetzt werden soll. Im ersten Schritt wird ein allfällig vorhandenes Label vermerkt. Sodann wird die Breite des Legendentextes in `\breite@opt` gespeichert. Wenn `breite@opt` leer ist, wird sie auf den Wert der Optionsvariable `\abblaenge@opt` (i.e. die Breite der Umgebung von `MultiAbb`) gesetzt. Nun wird der Legendentext mit `\erzeuge@leg@box` erzeugt. Weil diese Funktion die Legende in Bezug auf die Breite von `\bild@box` arbeitet, muss `\bild@box` eine `hbox` der Breite `\abblaenge@opt` sein (sinnvollerweise eine leere Box).

```
    \if@abbleg@%
        \edef\label@opt{\lk@label@opt}%
        \setze@label%
        \ifx\breite@opt\empty%
            \edef\breite@opt{\abblaenge@opt}
        \fi
        \setbox\bild@box=\hbox to\abblaenge@opt{}%
        \erzeuge@leg@box%
```

Nun wird die Legendenbox erzeugt. Sie ist eine `hbox` der Breite `\abblaenge@opt`. Je nach Wert von wird sie links- oder rechtsbündig oder zentriert gesetzt.

```
        \setbox\multileg@box=\hbox to\abblaenge@opt{%
            \if\legpos@opt l\else\hfill\fi%
            \box\leg@box%
            \if\legpos@opt r\else\hfill\fi%
            }%
```

Schliesslich wird `\@abbleg@` auf `false` gesetzt, damit die Figuren (innerhalb der Umgebung `MultiAbb` bei automatischer Numerierung keinen Legendentext erhalten. Aus dem gleichen Grund werden die Optionsvariablen für die automatische Numerierung mit `false` belegt. Die Optionsvariable `legpos` wird auf den ursprünglichen Wert zurückgesetzt.

```
        \global\@abbleg@false%
        \edef\lk@num@opt{false}%
        \edef\num@opt{false}%
        \setz@opts{legpos}%
    \fi%
```

Nun beginnt die eigentliche Umgebung `MultiAbb`. Das Material wird in einer `vbox` der Breite `\abblaenge@opt` gesetzt...

```
    \vbox\bgroup\hsize=\abblaenge@opt%
}%
```

`\endMultiAbb` ... und am Ende der Umgebung die Legende auf eine neue Zeile gesetzt. Schliesslich werden die Optionen auf ihre ursprünglichen Werte zurückgesetzt.

```
\long\def\endMultiAbb{%
    \egroup%
    \par%
    \vbox{\hsize=\abblaenge@opt\box\multileg@box}%
    \setz@opts{pos,ort,num,label,abst,breite,legpos,abblaenge}%
}%
```

`\opts@multiabb` Es folgen die Funktionen für die Setzungen der Optionen von `MultiAbb`. Das Argument `#1` ist eine Liste von Zuweisungen an Optionsvariablen. Dies Liste wird von `\m@filter@opt` rekursiv abgearbeitet. Falls `#1` leer ist, so wird der Vorgang mit `\m@flt@optii` abgebrochen. Andernfalls wird `\m@filter@opt` aufgerufen, wobei der Liste die Endmarkierung mit dem Token `\relax` hinzugefügt wird.

```
\def\opts@multiabb[#1]{%
    \edef\@arg{#1}%
    \ifx\@arg\empty%
        \let\next\m@flt@optii%
    \else%
```

```
        \let\next\m@filter@opt%
    \fi%
    \next[#1,\relax]
}%
```

`\m@filter@opt` Falls die Option unbekannt ist, wird sie ignoriert, un die Rekrsion fortgesetzt. Ansonsten werden den Optionen ihre lokalen Werte zugewiesen. Wenn das Listenende mit `\relax` erreicht ist, wird die Rekursion mit `\m@flt@optii` abgebrochen. Andernfalls wird sie via `\m@flt@optiii` fortgesetzt.

```
\def\m@filter@opt[#1=#2,#3]{%
    \@ifundefined{#1@opt}{\let\next\m@flt@optiii}{%
    \expandafter\xdef\csname lk@#1@opt\endcsname{#2}
    \expandafter\xdef\csname #1@opt\endcsname{#2}%
    \ifx#3\relax%
        \let\next\m@flt@optii%
    \else
        \let\next\m@flt@optiii
    \fi}%
    \next[#1=#2,#3]%
}%
```

`\m@flt@optii` `\m@flt@optiii` Die Funktion `\m@flt@optii` bewirkt den Abbruch der ekursion und `\m@flt@optiii` ruft `\m@filter@opt` mit dem Rest der Optionen-Liste auf.

```
\def\m@flt@optii[#1]{}%
\def\m@flt@optiii[#1=#2,#3]{%
    \m@filter@opt[#3]
}%
```

`\abb@fn` Es folgen die Funktionen für Fussnoten innerhalb der `AbbTxt`-Umgebung und im Legendentext.
Der numerische Variable dient als Zähler für die Fussnoten, damit mehr als eine Fussnaote möglich ist. Initial wird sie auf 0 gesetzt.

```
\newcount\abb@fn%
\abb@fn=0
```

`\AbbFootnote` Der LaTeX-Zähler `\@mpfn` enthält die aktuelle Fussnotenzahl. Dies wird um 1 erhöht. und mit dem LaTeX-Befehl `\thempfn` gesetzt. Sodann wird auch der Zähler `\abb@fn` um 1 erhöht. Schliesslich wird in der mit dem Zähler `\abb@fn` indizierten Variablen `Abb@fntxt`⟨*nummer*⟩ der Fussnotentext mit der Fussnotennummer ⟨*nummer*⟩gespeichert.

```
\def\AbbFootnote#1{%
    \addtocounter{\@mpfn}{1}%
    $^{\thempfn}$%
    \global\advance\abb@fn by1%
    \count1=\abb@fn%
    \expandafter\gdef\csname Abb@fntxt\number\count1\endcsname{#1}%
}%
```

`\AbbFootnote@` Diese Funktion setzt am Ende der `AbbTxt`-Umgebung oder nach der Setzung einer Legende die vorgängig gespeicherten Fussnoten. Zunächst ist der LaTeX-Zähler `\@mpfn` auf seinen ursprünglichen Wert zurückzusetzen.

```
\def\Abbfootnote@{%
    \count0=\abb@fn%
    \count1=1%
    \addtocounter{\@mpfn}{-\number\count0}%
```

In einer Schlaufe, deren Anzahl Durchläufe der Anzahl vorhandenen Fussnoten entspricht, werden die einschlägigen LaTeX-Varaiblen mit der Fussnotennummer und dem zugehörigen Fussnotentext belegt.

```
    \loop%
        \ifnum\count0>0
```

```
339         \addtocounter{\@mpfn}{1}%
340         \xdef\@thefnmark{\thempfn}%
341         \@footnotetext{\csname Abb@fntxt\number\count1\endcsname}%
342         \advance\count0 by-1%
343         \advance\count1by1%
344     \repeat%
```

Schliesslich wird der Zähler auf 0 zurückgesetzt.

```
345     \global\abb@fn=0%
346 }%

347 \catcode`\@=12
```

Index

Kursive Zahlen verweisen auf die Seite, auf welcher der Eintrag beschrieben ist, unterstrichene Zahlen verweisen auf die Definition, alle anderen auf die Verwendung.