

# L<sup>A</sup>T<sub>E</sub>X i tabelki

Grzegorz Sapijaszko

ul. Trzebnicka 6 m. 5, 50-246 Wrocław

grzegorz@sapijaszko.net

## Abstract

Tabelki w L<sup>A</sup>T<sub>E</sub>X-u... To temat nieśmiertelny. Ilość pytań na liście dyskusyjnej GUST-u oraz na grupie pl.comp.dtp.tex wskazuje na zainteresowanie tematem oraz na szereg wątpliwości, które pojawiają się przy definiowaniu tabel. W niniejszym artykule zaprezentowano podstawowe środowiska służące składaniu tabel, wraz z metodami wyrównywania tekstu w ich wnętrzu.

### Środowisko tabular

Jest to standardowe środowisko służące do składania tabel w L<sup>A</sup>T<sub>E</sub>X-u, posiadające jeden obowiązkowy parametr definiujący ilość i sposób justowania kolumn, np.:

```
\begin{tabular}{|c|l|r|}
```

```
...
```

```
\end{tabular}
```

spowoduje utworzenie tabeli o trzech kolumnach, pierwszej wyśrodkowanej (parametr c), drugiej dosuniętej do lewej (l) i trzeciej – do prawej (r). Pionowe linie (obramowania kolumn) tworzone są przy pomocy znaku |. W środowisku tym poszczególne wiersze tabeli rozdzielane są przy pomocy \\, poszczególne komórki wiersza znakiem &, zaś poziome linie oddzielające wiersze uzyskuje się poleceniem \hline. Aby uzyskać pustą komórkę w wierszu wystarczy wpisać & &. Przykładowo:

```
\begin{tabular}{|l|c|r|}
```

```
\hline
```

```
ala & ola & józia\\
```

```
\hline
```

```
janek & karol & zdzicho\\
```

```
grzesiek & czesiek & ktoś\\
```

```
\hline
```

```
\hline
```

```
ola & ania & zosia\\
```

```
\hline
```

```
\end{tabular}
```

spowoduje utworzenie tabelki przedstawionej poniżej:

|          |         |         |
|----------|---------|---------|
| ala      | ola     | józia   |
| janek    | karol   | zdzicho |
| grzesiek | czesiek | ktoś    |
| ola      | ania    | zosia   |

L<sup>A</sup>T<sub>E</sub>X sam zadba o wymaganą szerokość kolumn. Wszystko wygląda ładnie, dopóki tekst w ko-

mórkach jest dość krótki i cały wiersz mieści się na szerokości łamu. Problem zaczyna się, gdy istnieje potrzeba wpisania większej ilości tekstu do komórki, np.:

```
\begin{tabular}{|l|c|r|}
```

```
\hline
```

```
a1 & a2 & a3\\
```

```
\hline
```

```
b1 & b2 & b3\\
```

```
c1 & c2 & c3\\
```

```
\hline
```

```
\hline
```

```
długi tekst &
```

```
bardzo długi tekst &
```

```
bardzo długi tekst\\
```

```
\hline
```

```
\end{tabular}
```

co powoduje, iż wiersze w komórce nie są zawijane, lecz szerokość komórki (a zatem i tabeli) rozciągana jest tak, aby pomieścić cały tekst, wychodząc poza granice łamu:

|             |                    |                    |
|-------------|--------------------|--------------------|
| a1          | a2                 | a3                 |
| b1          | b2                 | b3                 |
| c1          | c2                 | c3                 |
| długi tekst | bardzo długi tekst | bardzo długi tekst |

Można temu zaradzić w dość prosty sposób, definiując szerokość kolumny. Zmodyfikujemy poprzedni przykład, np.:

```
\begin{tabular}{|lp{0.25\linewidth}|}
```

```
p{0.25\linewidth}|}
```

```
...
```

```
\end{tabular}
```

w rezultacie czego otrzymamy:

|             |                    |                                  |
|-------------|--------------------|----------------------------------|
| a1          | a2                 | a3                               |
| b1          | b2                 | b3                               |
| c1          | c2                 | c3                               |
| długi tekst | bardzo długi tekst | jeszcze jeden bardzo długi tekst |

Odległości między kolumnami można zmieniać korzystając z polecenia `@{...}`, które usuwa domyślny odstęp zastępując go wpisaniem między nawiasami. Polecenie to można wykorzystać do wyrównania zestawień (patrz też opis w rozdz. Pakiet `dcolumn`).

Poleceniem `\multicolumn` można łączyć komórki w wierszu, np.:

```
\begin{tabular}{|l|c|r|}
\hline
a1 & a2 & a3\\
\hline
\multicolumn{3}{|c|}{b1b2b3}\\
\hline
c1 & \multicolumn{2}{r|}{c2c3}\\
\hline
d1 & d2 & d3\\
\hline
\end{tabular}
```

po przekompilowaniu otrzymuje się:

|        |      |    |
|--------|------|----|
| a1     | a2   | a3 |
| b1b2b3 |      |    |
| c1     | c2c3 |    |
| d1     | d2   | d3 |

Jako parametry polecenia `\multicolumn` podaje się kolejno: ilość komórek wiersza, które mają zostać złączone, wyrównanie i ewentualnie obramowanie oraz zawartość. Polecenie to można wykorzystać również do uzyskania wyrównania w pojedynczej komórce wiersza innego, niż zadeklarowano dla całej kolumny, np.:

```
\multicolumn{1}{r}{do prawa}
```

Łączenie komórek w obrębie jednej kolumny (czyli w pionie) można zrealizować przy pomocy polecenia `\multirow` z pakietu `multirow`. Najprostszy zapis polecenia może wyglądać w sposób następujący:

```
\multirow{ilość wierszy}
{szerokość}{zawartość}
```

Lekko modyfikując poprzedni przykład:

```
\begin{tabular}{|l|c|r|}
\hline
a1 & a2 & a3\\
\hline
\multicolumn{3}{|c|}{b1b2b3}\\
\hline
\end{tabular}
```

```
\hline
\multirow{2}{1cm}{c1d1} &
\multicolumn{2}{r|}{c2c3}\\
\cline{2-3}
& d2 & d3\\
\hline
\end{tabular}
```

otrzymujemy:

|        |      |    |
|--------|------|----|
| a1     | a2   | a3 |
| b1b2b3 |      |    |
| c1d1   | c2c3 |    |
|        | d2   | d3 |

Należy zwrócić uwagę na wprowadzone polecenie `\cline{m-n}`, które tworzy linię poziomą od komórki `m` do komórki `n`.

### Pakiet `tabularx`

Elegantszym rozwiązaniem problemu z żadaną szerokością tabeli jest użycie pakietu `tabularx`, który w przeciwieństwie do środowiska `tabular` modyfikując szerokości poszczególnych kolumn pozwala na zdefiniowanie tabeli o określonej szerokości:

```
\begin{tabularx}{250pt}{kolumny}
...
\end{tabularx}
```

Dla przykładu zdefiniowano tabelkę o szerokości równej szerokości linii:

```
\begin{tabularx}{\linewidth}{|r|X|l|X|}
...
\end{tabularx}
```

co w rezultacie daje:

|              |  |      |  |
|--------------|--|------|--|
| dwie komórki |  | ala  | ola  |
| jaś          | bardzo długi tekst, który powinien się pozawijać w dość szerokiej kolumnie | staś | i jeszcze jedna szeroka kolumna, o szerokości takiej samej jak kolumna druga |

Kolumny, szerokością których steruje środowisko `tabularx` powinno być określone jako `X`. Niestety, efekty wizualne nie są najładniejsze – nie jest przeprowadzana optymalizacja kar za łamanie tekstu w każdej z kolumn, a kolumny „sterowane” są tej samej szerokości.

Można z tym próbować walczyć wprowadzając wyrównanie do lewej w którejś z kolumn poleceniem `\raggedright`, jednak efekt nadal nie jest najlepszy:

|              |  |      |  |
|--------------|--|------|--|
| dwie komórki |  | ala  | ola  |
| jaś          | bardzo długi tekst, który powinien się pozawijać w dość szerokiej kolumnie | staś | i jeszcze jedna szeroka kolumna, o szerokości takiej samej jak kolumna druga |

### Pakiet array

Pakiet ten umożliwia zdefiniowanie typu kolumny, którą następnie można wykorzystać w środowiskach `tabular`, `tabularx`, `longtable` oraz `supertabular`. Kolumnę definiuje się następująco:

```
\newcolumntype{nazwa}[n]{specyfikacja}
```

gdzie `nazwa` to nazwa używana później w definicji tabeli, `n` – liczba opcjonalnych parametrów, zaś `specyfikacja` zawiera definicję kolumny. W definicji tej mogą być użyte znane już polecenia dosunięcia tekstu do jednej lub drugiej strony oraz centrowania (odpowiednio `l`, `r`, `c`), można też zdefiniować kolumnę o określonej szerokości (poleceniem `m{szer}`). Polecenie `p{szer}` tworzy kolumnę o zadanej szerokości zaś zawartość brana jest w pudełko równoważne poleceniu `\parbox[t]{szer}` (pozycjonowane wg szczytu komórki). Podobnie, polecenie `b{szer}` odpowiada poleceniu `\parbox[b]{szer}` – pozycjonowanie wg dołu komórki. Niżej podano przykład tabeli, której dwie kolumny zdefiniowano przy użyciu polecenia `\newcolumntype`:

```
\newcolumntype{j}{m{50pt}}
\newcolumntype{d}{p{30pt}}
```

```
\begin{tabular}{|c|j|r|d|}
...
\end{tabular}
```

|       |  |     |                                 |
|-------|--|-----|---------------------------------|
| ala   | ciekawa kolumna o szerokości 50pt no i jeszcze trochę tekstu | ola | coś tam innego dłuższego że hej |
| jasio | 1  | 2   | stasio                          |

W definicji kolumny można używać również makrodefinicji, które będą wykorzystywane przed (`>`) lub po (`<`) zawartości komórki, np. definicja:

```
>{\bfseries}r
```

utworzy kolumnę, której zawartość komórek będzie dosunięta do prawej strony i jednocześnie pogrubiona, co ilustruje poniższy przykład.

|       |                |        |
|-------|----------------|--------|
| ala   | <b>zosia</b>   | ola    |
| jasio | <b>krzysio</b> | stasio |

Podobnym makrem można zdefiniować kolumnę o stałej szerokości, jednak z tekstem dosuniętym do prawej strony:

```
>{\raggedleft}p{5cm}
```

Polecenia wpływające na wizualny wygląd poszczególnych komórek to m.in.:

`\extrarowheight` – zwiększa lub zmniejsza standardową wysokość każdego wiersza w tabeli o podany wymiar, np.:

```
\setlength{\extrarowheight}{2pt}
```

`\arraycolsep` – połowa szerokości odstępu między kolumnami w trybie matematycznym,

`\tabcolsep` – połowa szerokości odstępu między kolumnami,

`\arrayrulewidth` – szerokość pionowej linii (krawędzi) kolumny,

`\doublerulesep` – odległość między dwoma pionowymi liniami,

### Pakiet blkarray

Pakiet zawiera polecenia analogiczne jak pakiety `hline`, `array` oraz `multirow`. Zdefiniowane jest w nim środowisko `blockarray` zastępujące środowiska `tabular` i `array` – w trybie matematycznym zachowuje się jak `array`, zaś w tekstowym jak `tabular`. Pakiet ten oferuje też szereg predefiniowanych linii rozdzielających poszczególne komórki. Służy do tego celu polecenie `\BAhhline{parametry}`, gdzie jako `parametry` mogą wystąpić:

- = podwójna linia o szerokości kolumny
- ” podwójna linia przerywana o szerokości kolumny
- linia pojedyncza
- . pojedyncza linia przerywana
- ~ brak linii w kolumnie
- | linia pionowa przecinająca linie poziome
- : linia pionowa stykająca się z podwójną linią poziomą
- # skrzyżowanie podwójnych linii
- t wykończenie rogu górnego
- b wykończenie rogu dolnego
- \* powtórzenie sekwencji, np. `*{3}{==#}` odpowiada `==#==#==#`

Przykładowo zapis:

```
\begin{blockarray}{|cc||c|c|}
\BAhhline{|t:::t:::t|}
a & b & c & d\\
\BAhhline{||:::|~|~|}
1 & 2 & 3 & 4\\
\BAhhline{#==#~|#}
```

```
e & f & g & h \\
\BAhhline{||--||--||}
5 & 6 & 7 & 8 \\
\BAhhline{||==||==||}
i & j & k & l \\
\BAhhline{||==:|=|=}
9 & 10 & 11 & 12 \\
\BAhhline{||=":"|=|=}
m & n & o & p \\
\BAhhline{||-|.||-||}
13 & 14 & 15 & 16 \\
\BAhhline{|b==:b==:b|}
\end{blockarray}
```

tworzy poniższą tabelkę:

|    |    |    |    |
|----|----|----|----|
| a  | b  | c  | d  |
| 1  | 2  | 3  | 4  |
| e  | f  | g  | h  |
| 5  | 6  | 7  | 8  |
| i  | j  | k  | l  |
| 9  | 10 | 11 | 12 |
| m  | n  | o  | p  |
| 13 | 14 | 15 | 16 |

Dokładny opis pakietu podał Włodzimierz Macewicz w [1].

### Pakiet dcolumn

Pakiet dcolumn wprowadza polecenie D, które może być argumentem polecenia \newcolumntype, np.:

```
\newcolumntype{nazwa}
{D{tsep}{dsep}{ilość miejsc}}
```

gdzie tsep określa separator używany w tekście źródłowym (może być to kropka lub przecinek), dsep to znak jaki będzie wstawiany przez TEX-a w miejsce tsep na wydruku, ilość miejsc określa wyrównanie zawartości komórki względem kolumny: wartość -1 oznacza, że znak separatora dziesiętnego znajdzie się w środku kolumny, zaś wartość dodatnia oznacza, ile miejsc dziesiętnych ma być przeznaczonych na część ułamkową.

### Przypisy w tabelach

Przypisów w komórkach tabeli nie da się wprowadzić w standardowy sposób, wykorzystując polecenie \footnote – co prawda pojawi się numer przypisu, ale nie pojawi się treść. Ograniczenie to można obejść wykorzystując polecenia \footnotemark i \footnotetext, np.:

```
\begin{tabular}{llll}
a1 & a2\footnotemark & a3 \\
b1\footnotemark & b2 & b3\footnotemark \\
\end{tabular}
```

```
\end{tabular}
```

a następnie, zaraz po zakończeniu środowiska tabular, należy wpisać treści odnośników, pamiętając o dokładnym policzeniu ilości przypisów i odpowiedniego ustawienia licznika:

```
a1 & a2\footnotemark & a3 \\
b1\footnotemark & b2 & b3\footnotemark \\
\end{tabular}
\addtocounter{footnote}{diff}
\footnotetext{pierwszy odnośnik}
\stepcounter{footnote}
\footnotetext{drugi odnośnik}
\stepcounter{footnote}
\footnotetext{trzeci odnośnik}
```

gdzie diff to ujemny parametr, którego bezwzględna wartość to ilość odwołań minus jeden. W powyższym przykładzie diff powinna wynosić -2. W rezultacie otrzymuje się:

$$\begin{matrix} a1 & a2^1 & a3 \\ b1^2 & b2 & b3^3 \end{matrix}$$

Inną możliwością jest skorzystanie z pakietu threeparttable, który umożliwia używanie odsyłaczy wewnątrz tabeli co ilustruje poniższy przykład:

```
\begin{threeparttable}
\begin{tabular}{...}
ala\tnote{1} & ola \\
stasio & jasio\tnote{2} \\
\end{tabular}
\end{threeparttable}
\begin{tablenotes}
\item [1] pierwszy przypis
\item [2] drugi przypis
\end{tablenotes}
\end{threeparttable}
```

w rezultacie kompilacji powstaje:

|                         |                    |
|-------------------------|--------------------|
| ala ma asa <sup>1</sup> | ola ma kota        |
| stasio                  | jasio <sup>2</sup> |

<sup>1</sup> pierwszy przypis

<sup>2</sup> drugi przypis

c.d.n

### Bibliografia

[1] Włodzimierz Macewicz. Pakiet blkarray. *Biuletyn GUST*, 14:27–34, 2000. Wersja elektroniczna dostępna jest pod adresem <ftp://ftp.gust.org.pl/TeX/GUST/bulletin/14/04wm.ps.gz>.

<sup>1</sup> pierwszy odnośnik

<sup>2</sup> drugi odnośnik

<sup>3</sup> trzeci odnośnik