Tab – A Lout Package for Formatting Tables

Jeffrey H. Kingston

Basser Department of Computer Science
University of Sydney 2006
Australia

ABSTRACT

This report describes the use of Tab, a package of definitions for producing tables with the Lout document formatter:

Value of mathematical formulae (millions of dollars)		
Quadratic formula	$x = \dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$	3.5
Binomial theorem	$(a+b)^n = \sum_{k=0}^{\infty} \binom{n}{k} a^k b^{n-k}$	12

A wide variety of styles for columns and individual entries is provided, and entries may contain equations, figures, and other objects without restriction.

22 December 1992

Tab – A Lout Package for Formatting Tables

Jeffrey H. Kingston

Basser Department of Computer Science
University of Sydney 2006
Australia

1. Introduction

The Tab table formatting package is designed to simplify the production of tables in documents produced with the Lout document formatting system [1]. Tab aligns entries in columns, draws boxes and rules, and so on. To use Tab, either place -itab on the Unix[1] command line, or @SysInclude { tab } in the setup file.

To begin with a very simple example, the table

Austen	Chaucer	Donne
Balzac	Darwin	Goethe
Byron	Dickens	Homer

is the result of the following Lout input:

```
@Tab
   @Fmta { @Col A  !  @Col B  !  @Col C }
{
   @Rowa A { Austen } B { Chaucer } C { Donne }
   @Rowa A { Balzac } B { Darwin } C { Goethe }
   @Rowa A { Byron } B { Dickens } C { Homer }
}
```

Immediately after the @Tab symbol, which introduces the table, comes a *format option*, @Fmta, describing the format of each row. It says that each row contains three columns: @Col A, @Col B, and @Col C. The format option may have up to 26 columns, with names chosen freely from the upper-case letters from A to Z. The symbol ! separates each pair of columns.

After the format option comes the body of the table, enclosed in braces. It consists entirely of a sequence of rows, each introduced by the @Rowa symbol and containing one entry for each column of the format option, as shown (the row may occupy any number of lines of the input file). The entries may be arbitrary Lout objects, such as words, paragraphs, equations, figures, and so on without restriction. An entry may be omitted altogether if it is empty. Lout will choose suitable widths for the columns, and break paragraphs in the entries to the right widths.

Tab neither knows nor cares what happens to the table it produces; it could appear at any point in the document, even within a paragraph of text. Most commonly, though, tables are displayed using the @IndentedDisplay and @CentredDisplay symbols from the DocumentLayout package [2]:

[1]Unix is a trademark of AT&T Bell Laboratories.

```
@CentredDisplay @Tab ...
```

or else they go into the @Table symbol:

```
@Table
   @Caption { ... }
@Tab
{ ...
}
```

which centres them at the top of the following page and adds a caption.

2. Changing the style of entries and columns

The style of an entry is mainly determined by four symbols from standard Lout: @Font, @Break, @Wide, and @High. We will review them only briefly here, since they are treated elsewhere [1, 2], then explain how they are used within tables.

The @Font symbol changes the font and font size of an object. For example,

```
Slope @Font { hello world }
```

produces *hello world*, and

```
{ Bold +2p } @Font { hello world }
```

produces

hello world

which is two points larger than it otherwise would have been, as well as set in bold. The DocumentLayout package provides the abbreviations @I and @B for Slope @Font and Bold @Font.

The @Break symbol affects paragraph breaking. Most relevant here are clines @Break and rlines @Break, which will centre or right-justify one or more lines in the given column:

```
clines @Break {
A small centred
paragraph.
}
```

produces

A small centred
paragraph.

for example. Actually, there is a small problem with these symbols: they only work on paragraphs of at least two words, so they will fail to centre or right-justify a solitary word. Tab provides two symbols, @CC and @RR, which do essentially the same thing, but which work on single words as well.

The @Wide symbol forces an object to have a given width, by padding it with white space on the right if it is too small, or breaking its paragraphs if it is too wide:

```
1.5i @Wide {
This small paragraph will
be broken to a width of
one and one half inches.
}
```

produces

> This small paragraph will be broken to a width of one and one half inches.

Widths may be specified in centimetres, points or ems using the letters c, p and e instead of i. The @High symbol is very similar, forcing an object to have a given height by adding white space beneath it.

Although we will use only these four symbols in the following, more exotic symbols may be used, for example 90d @Rotate which rotates any object (the entire table, for example, or one entry) ninety degrees, or any other positive or negative angle. Symbols defined by the user or taken from other packages may also be used.

To change the style of an individual entry, enclose it in the symbols required:

```
@Rowa
   A { @CC { one entry of the table } }
```

More commonly, the same style is to be applied to every entry in a particular column, and this is done by placing the symbols for the style inside the format option, between @Col and the column letter:

```
@Tab
   @Fmta { @Col 3c @Wide @CC A  !  ... }
```

Any @CC or @RR symbol should appear last, immediately before the column letter. Now every entry in column A will be centred in a three centimetre space, which means that the column as a whole will be three centimetres wide (excluding margins) and its entries will be centred.

It is best not to use the @Wide symbol at first, since Lout is quite good at choosing appropriate column widths. If the result is not satisfactory, and the table is wide enough to require paragraph breaking within its entries, @Wide should be used only to reduce the width of the wider columns, not to increase the width of the narrower ones.

3. Changing the style of rows

Sometimes different rows need different formats, usually because they are headings. Although this can be done with style-changing symbols in the entries, it is probably better to use multiple format options:

```
@Tab
   @Fmta { @Col @Heading A  !  @Col @Heading B }
   @Fmtb { @Col @A  !  @Col @CC B }
{
   @Rowa A { Course } B { Enrolment }
   @Rowb A { Software Engineering } B { 174 }
   @Rowb A { Complexity Theory } B { 37 }
}
```

has result

Course	**Enrolment**
Software Engineering	174
Complexity Theory	37

We have used the @Heading symbol from the DocumentLayout package just to prove that it can be done. Each row appears in the format of the corresponding format option: @Rowa rows in the format of @Fmta, @Rowb rows in the format of @Fmtb, and so on. There may be up to ten different format options, from @Fmta to @Fmtj, and they may be used repeatedly and in any order.

There is an @Over symbol for constructing *spanning columns*: columns that spread over two or more following columns. For example, the Lout input

```
@I @Tab
   @Fmta { @Col @CC X @Over A,B  !  @Col C }
   @Fmtb { @Col A  !  @Col B  !  @Col C }
{
   @Rowa X { X } C { C }
   @Rowb A { A } B { B } C { C }
}
```

produces the table

<table>
<tr><td colspan="2" align="center">X</td><td align="center">C</td></tr>
<tr><td align="center">A</td><td align="center">B</td><td align="center">C</td></tr>
</table>

(We will see how to get the lines later.) The @Over symbol always comes directly after the column letter, and it is followed by one or more letters separated by commas indicating which columns in the following row this column is to span over.

With multiple row formats, especially those with spanning columns, it is important to take care that each row is compatible with the one preceding it, in the sense of having the same number of columns, taking @Over symbols into account. For example, two successive @Rowa rows would not be compatible in the example above, since the second has only two columns but the first prepares the way for three.

Tab does not provide a way to have spanning columns underneath the columns they span, so spanning columns are useful only for headings. This restriction was made necessary by inadequacies in the way that the Basser Lout interpreter handles spanning columns.

4. Aligned columns

Columns of numbers are often presented with decimal points aligned:

```
  5.46
128.2
  3.4159
5772
```

This is achieved with Tab by placing the ^ symbol (which actually comes from the DocumentLayout package) just before the alignment point in each entry:

```
@Tab
   @Fmta { @Col A }
{
   @Rowa A { 5^.46 }
   @Rowa A { 128^.2 }
   @Rowa A { 3^.4159 }
   @Rowa A { 5772^ }
}
```

The equals signs of equations can be aligned in the same way. The format of such columns should not contain @CC or @RR.

The simplest way to get a heading over an aligned column is to use the ^ symbol in the heading as well. If it is not possible to do this, for example because the heading entry is a paragraph of text broken over more than one line, the most viable alternative is to use @Over in the heading line's format to make the heading column span over the aligned column:

```
@Tab
   @Fmta { @Col @CC H @Over A }
   @Fmtb { @Col A }
{
   @Rowa H { Heading }
   @Rowb A { 5^.46 }
   @Rowb A { 128^.2 }
   @Rowb A { 3^.4159 }
   @Rowb A { 5772^ }
}
```

for example produces

```
 Heading
  5.46
128.2
  3.4159
5772
```

Unfortunately, the aligned entries are left-justified in the column, not centred.

5. Margins

Tab places a *vertical margin* above and below each row, and a *horizontal margin* to the left and right of each column. (If there are no side rules, the leftmost and rightmost margins are omitted.) The size of these margins is determined by two options to the @Tab symbol:

```
@Tab
   hmargin { 0.3c }
   vmargin { 0.2v }
   @Fmta { ... }
...
```

Shown here are the default values, 0.3 centimetres for the horizontal margins, and 0.2 times the current inter-line spacing (denoted by the letter v) for the vertical margins.

The horizontal margin is fixed, but each row may have its own vertical margin:

```
@Rowa
   vmargin { 0.5vx }
   A { ... }
```

If omitted, the vmargin value is taken from the @Tab option.

The value shown here, 0.5vx, requests a vertical margin of half the current inter-line spacing, measured from the baseline of the entry rather than from its edge (this is what the x means). This value is a good choice when the entries contain simple words or lines of text (i.e. no paragraphs, large equations, figures, etc.):

Course	**Enrolment**
Software Engineering	174
Complexity Theory	37

since the lines of the table will then be separated evenly like the lines in paragraphs.

6. Rules, boxes, multi-page tables, etc.

Tab draws four kinds of rules. *Above rules* go across the table above a specified row; *below rules* go across the table below a row. *Between rules* go down the table between any two columns, but not before the first or after the last; and *side rules* go down the two sides:

```
@Tab
   above { no }
   below { yes }
   between { single }
   side { double }
   @Fmta { ... }
...
```

Each option may have the value no (no rules), yes or single (draw single rules), or double (draw double rules). Omitting an option is equivalent to giving it the value no. A rule below one row exactly overstrikes a rule above the next, unless a page break intervenes.

These options to @Tab will draw all possible rules of the given type. If only some are wanted, there are above and below options (but no side option) for each row symbol:

```
@Rowf
   above { yes }
   below { double }
   A { ... }
   ...
```

and there is a between option for the ! symbol that separates columns in the format options:

```
@Col A  ! between { double }  @Col B
```

The symbols !! and !!! are abbreviations for ! between { single } and ! between { double }. All these options override the corresponding @Tab options when they are present; when they are absent, the @Tab options hold sway.

It is not hard to assemble a single box around a table, using side rules, a rule above the first row, and a rule below the last:

```
@Tab
   side { single }
{
   @Rowa above { single } ...
   ...
   @Rowj below { single } ...
}
```

Double boxes around a table should be assembled using the following recipe:

```
@Tab
   side { double }
{
   @Rowa above { doubleboxabove } ...
   ...
   @Rowj below { doubleboxbelow } ...
}
```

This is a little tedious, but the result is good.

The symbols @Line and @DoubleLine produce horizontal single and double lines of the width of the column. Two other symbols, @Rule and @DoubleRule, produce lines which extend into the adjacent margins as well. For example,

5.46
3.4159
5772
5909.0759

is produced by

```
@Tab
   @Fmta { @Col A }
{
   @Rowa A { 5^.46 }
   @Rowa A { 3^.4159 }
   @Rowa A { 5772^ }
   @Rowa A { @Line } vmargin { 0i }
   @Rowa A { 5909^.0759 }
}
```

Owing to problems behind the scenes, @Rule and @DoubleRule fail in aligned columns.

The @Tab symbol has a linewidth option which determines the thickness of all the lines:

```
@Tab
   linewidth { 0.5 pt }
```

shows the default value, 0.5 points. This cannot be overridden within the table. Double rules are drawn with their centres three line widths apart. Tab thinks that rules and double rules occupy no space, so thick rules may require larger margins.

The tables produced by Tab permit page breaks between any two rows. The choice of page breaks can either be left to Lout, or it can be forced by placing the new page symbol @NP between two rows. To ensure that each part of a multi-page table is boxed, the rows preceding the page breaks must have below rules, and the rows following them must have above rules. The @IndentedDisplay and @CentredDisplay symbols of the DocumentLayout package prevent page breaks in the thing displayed, so they cannot be used to display multi-page tables; a multi-page table should be displayed at the left margin by preceding and following it with @DP. To prevent page breaks within a small table, either use one of the display symbols just mentioned or else use @OneRow @Tab ....

The @HAdjust symbol from standard Lout can be used to widen a table's columns until it occupies the full column width. The recommended arrangement is

```
@DP
@HAdjust @Tab
  ...
@DP
```

since @HAdjust is not effective within displays.

Expert users might occasionally find a use for the @MarkRow symbols. There is one of them for each @Row symbol (@MarkRowa for @Rowa, etc.). A @MarkRow symbol causes the principal mark of the table to pass through its row, permitting that row to be aligned with horizontally adjacent objects.

7. A final example

In conclusion, here is the input for a relatively large table:

```
@OneRow 10p @Font 1.15fx @Break @Tab
   linewidth { 1.0 pt }
   vmargin { 0.4v }
   @Fmta { @Col @B A  !  @Col @B B  !  @Col @B C  !  @Col @B D }
   @Fmtb { @Col outdent @Break A  !  @Col 1.8c @Wide ragged @Break B !
         @Col C  !  @Col outdent @Break D }
{

@Rowa above { single } below { single }
   A { Scholarship }
   B { Value $ }
   C { Closing date }
   D { Qualifications }

@Rowa
   A { 1.  Tenable at the University of Sydney }

@Rowb
   A { Commonwealth and University Postgraduate Research Awards }
   B { 10,415 (12,724 to 16,433 from 1 January 1990) }
   C { 31 October }
   D { Graduates with Hons I or Hons II Div. 1.  For research in any field }

@Rowb
   A { Commonwealth Postgraduate Course Awards }
   B { 10,415 }
   C { 30 September }
   D { Graduates with honours degrees or very good pass degrees.  For
Master's degrees undertaken by coursework }

@Rowb
   A { R. and M. Bentwich Scholarship }
   D { Graduate who holds a postgraduate research scholarship and who
requires a supplementary grant }

@Rowa
   A { 2.  Awarded by external bodies }

@Rowb
   A { Caltex }
   B { 24,000 }
   C { 30 September }
   D { Female graduates completing degree or diploma in year of application }

@Rowb
   A { Rhodes Scholarship }
   B { @Sterling3500 + return air-fare }
   C { 1 October }
   D { Age limit 25.  For tenure at the University of Oxford }
   below { single }
}
@DP
10p @Font { *Additional benefits include cost of travel and payment of fees. }
```

The most interesting things about it are the use of outdented paragraphs, and the @Wide symbol which restricts the width of the second column, which would otherwise be as wide as the first and last. Here is the result:

Scholarship	Value $	Closing date	Qualifications
1. Tenable at the University of Sydney			
Commonwealth and University Postgraduate Research Awards	10,415 (12,724 to 16,433 from 1 January 1990)	31 October	Graduates with Hons I or Hons II Div. 1. For research in any field
Commonwealth Postgraduate Course Awards	10,415	30 September	Graduates with honours degrees or very good pass degrees. For Master's degrees undertaken by coursework
R. and M. Bentwich Scholarship			Graduate who holds a postgraduate research scholarship and who requires a supplementary grant
2. Awarded by external bodies			
Caltex	24,000	30 September	Female graduates completing degree or diploma in year of application
Rhodes Scholarship	£3500 + return air-fare	1 October	Age limit 25. For tenure at the University of Oxford

*Additional benefits include cost of travel and payment of fees.

References

1. Kingston, Jeffrey H., *Document Formatting with Lout (Second Edition)*. Tech. Rep. 449 (1992), Basser Department of Computer Science F09, University of Sydney 2006, Australia.

2. Kingston, Jeffrey H., *A beginners' guide to Lout*. Tech. Rep. 450 (1992), Basser Department of Computer Science F09, University of Sydney 2006, Australia.