A Practical Introduction to the

Lout

Document Formatting System

Jeffrey H. Kingston

Basser Dept. of Computer Science
The University of Sydney

A simple input file

```
@SysInclude { doc }
@Doc @Text @Begin
Hello, world
@End @Text
```

How to format it

```
lout filename > out.ps
ghostview out.ps
mpr out.ps
```

Hello, world

Headings and paragraphs

```
@SysInclude { doc }
@Doc @Text @Begin
@Heading { Introduction }
@PP
The design of the Lout formatting
system was undertaken with the
needs of the @I { ordinary user }
very much in mind.
@End @Text
```

Introduction

The design of the Lout formatting system was undertaken with the needs of the *ordinary user* very much in mind.

Displays

You certainly don’t want to return to his office and report:
@IndentedDisplay @I {
‘I can’t find an efficient algorithm, I guess I’m just too dumb.’
}
To avoid serious damage to your position in the company, it would be better if ...

You certainly don’t want to return to his office and report:

> *‘I can’t find an efficient algorithm, I guess I’m just too dumb.’*

To avoid serious damage to your position in the company, it would be better if …

Paragraph breaking styles

You certainly don’t want to return to
his office and report:
@ID { ragged nohyphen } @Break @I {
‘I can’t find an efficient algorithm, I
guess I’m just too dumb.’
}
To avoid serious damage to your
position in the company, it would
be better if ...

You certainly don't want to return to his office and report:

> *'I can't find an efficient algorithm, I guess I'm just too dumb.'*

To avoid serious damage to your position in the company, it would be better if …

Lists

```
@Heading { Operating Instructions }
@NumberedList
@ListItem { Press small green lever. }
@ListItem { Wait approximately 10 seconds
until red light flashes. }
@ListItem { If smoke emerges from rear of unit,
call Service Department. }
@EndList
```

Operating Instructions

1. Press small green lever.

2. Wait approximately 10 seconds until red light flashes.

3. If smoke emerges from rear of unit, call Service Department.

Technical reports

```
@SysInclude { report }
@Report
    @Title { ... }
    @Author { ... }
    @Institution { ... }
    @DateLine { ... }
//
@Abstract { ... }
@Section { ... }
@Section { ... }
@Section { ... }
@Appendix { ... }
@Appendix { ... }
```

Sections

```
@Section
   @Tag { dfs }
   @Title { Depth-first search }
@Begin
@PP
We turn now to our first algorithm
on general graphs ...
@End @Section
```

10.6. Depth-first search

We turn now to our first algorithm on general graphs …

Cross references

For further information, consult Section @NumberOf dfs on page @PageOf { dfs }.

For further information, consult Section 10.6 on page 245.

References

@Database @Reference { myrefs }
...
For the details, consult the User's
Guide @Cite { $kingston1995lout.user }.

For the details, consult the User's Guide [1].

…

References

1. Jeffrey H. Kingston. *A User's Guide to the Lout Document Formatting System (Version 3).* Basser Department of Computer Science, University of Sydney, 1995.

2. …

Database file myrefs.ld

```
{ @Reference
    @Tag { kingston1995lout.user }
    @Type { Book }
    @Author { Jeffrey H. Kingston }
    @Title { A User’s Guide to the Lout
Document Formatting System (Version 3) }
    @Institution { Basser Department of
Computer Science }
    @Address { University of Sydney
2006, Australia }
    @Year { 1994 }
}
```

Books (and theses)

- Title page, preface, introduction
- Automatic table of contents
- Prefatory pages numbered in Roman numerals
- Chapters, sections, subsections, appendices
- References at end of chapters or book
- Running page headers
- Odd-even page formats
- Sorted index

Making a sorted index

@PP
There are several possible ways to implement the
@I Partition procedure,
partition @Index { @I Partition (in {@I Quicksort}) }
but the following seems to be the best. Starting ...

Index

…
partial order, 227
Partition (in *Quicksort*), 189
postorder traversal
  of binary tree, 19
  topological ordering, 229
…

Equation formatting

```
@SysInclude { eq }
...
Since @Eq { T(n-i) = T(0) = 0 } we have
@IndentedDisplay @Eq {
T(n) = big sum from i=0 to n-1 2 sup i = 2 sup n - 1
}
for the number of disk moves made by the Towers
of Hanoi algorithm, given @Eq { n } disks.
```

Since $T(n - i) = T(0) = 0$ we have

$$T(n) = \sum_{i=0}^{n-1} 2^i = 2^n - 1$$

for the number of disk moves made by the Towers of Hanoi algorithm, given n disks.

Another equation

```
@CenteredDisplay @Eq {
big int supp 1 on 0 ‘
dx over sqrt { 1 - x sup 2 }
= pi over 2
}
```

$$\int_0^1 \frac{dx}{\sqrt{1 - x^2}} = \frac{\pi}{2}$$

Tables

```
@SysInclude { tab }
...
@Tab
    @Fmta { @Col @I A ! @Col B }
{
@Rowa
   A { Fortran }
   B { The first ... language }
@Rowa
   A { Algol-60 }
   B { Said to be ... successors }
@Rowa
   A { Pascal }
   B { The famous ... successors }
}
```

Fortran	The first high-level programming language
Algol-60	Said to be a better language than most of its successors
Pascal	The most famous of Algol-60's successors

Another table

```
@Tab
  hmargin { 0.4c }
  vmargin { 0.3v }
  side { single }
  @Fmta { @Col @B @CC X @Over A,B,C }
  @Fmtb { @Col @I A ! @Col B !! @Col C }
{

@Rowa  above { single }
  X { Value of mathematical ... dollars) }

@Rowb  above { double }
  A { Quadratic formula }
  B { @Eq { x ^= { ... } over 2a } }
  C { 3^.5 }

@Rowb  below { single }
  A { Binomial theorem }
  B { @Eq { ( a + b ) sup n ^= ... b sup n-k } }
  C { 12^ }
}
```

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

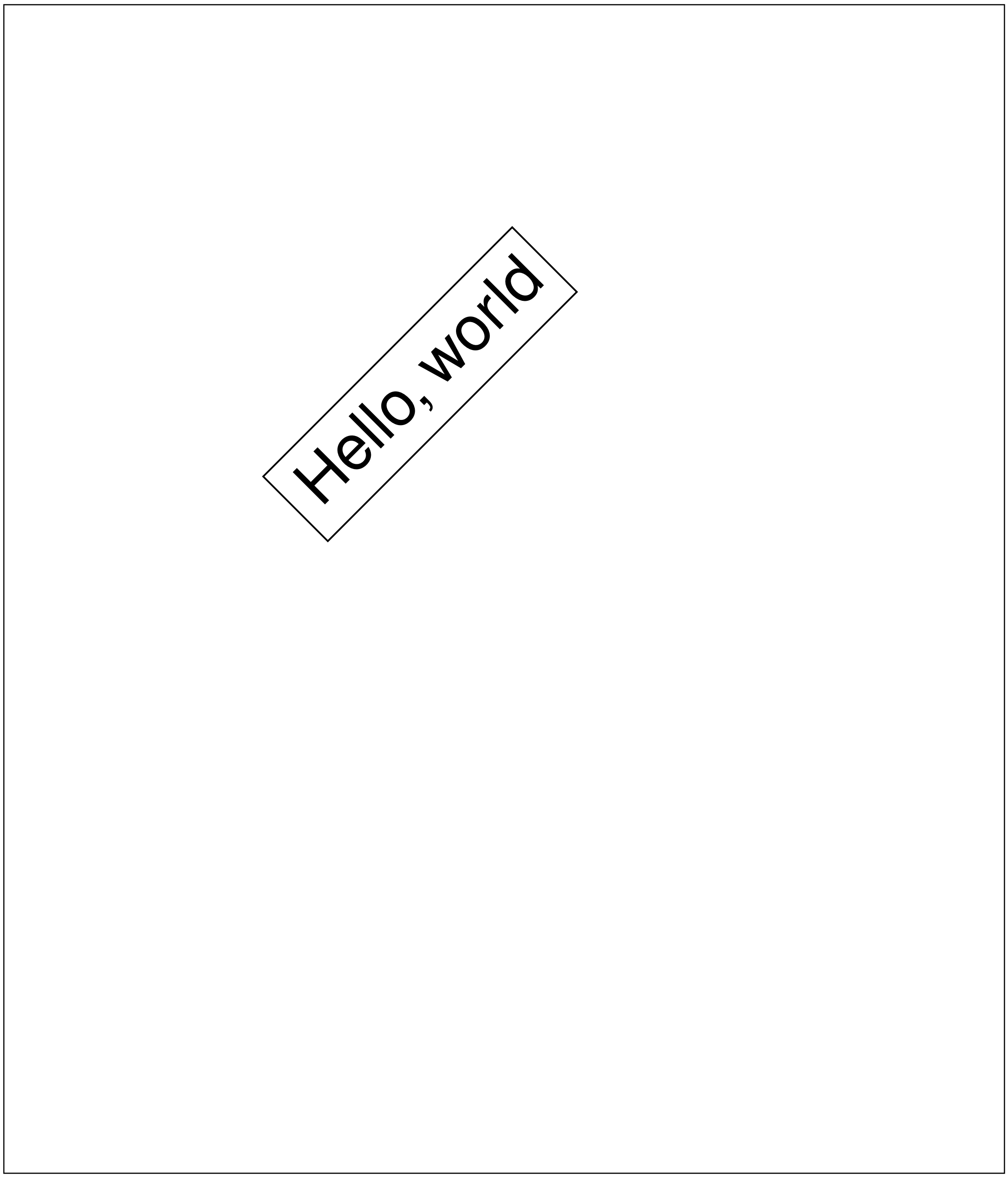
Pascal programs

```
@SysInclude { pas }
...
@ID @Pas {
procedure DoPriAbstract(root: PriEntry);
begin
   if root^.leftchild <> nil then begin
      DoPriAbstract(root^.leftchild);
      write(', ');
   end;
   PriKeyAbstract(root^.key);
   write(':');
   PriValueAbstract(root^.value);
   if root^.rightchild <> nil then begin
      write(', ');
      DoPriAbstract(root^.rightchild);
   end;
end;
}
```

```
procedure DoPriAbstract(root: PriEntry);
begin
   if root↑.leftchild ≠ nil then begin
      DoPriAbstract(root↑.leftchild);
      write(', ');
   end;
   PriKeyAbstract(root↑.key);
   write(':');
   PriValueAbstract(root↑.value);
   if root↑.rightchild ≠ nil then begin
      write(', ');
      DoPriAbstract(root↑.rightchild);
   end;
end;
```

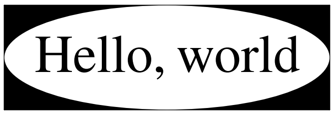
Basic graphics

```
45d @Rotate 1.5 @Scale @Box {
    Hello, world
}
```


Hello, world

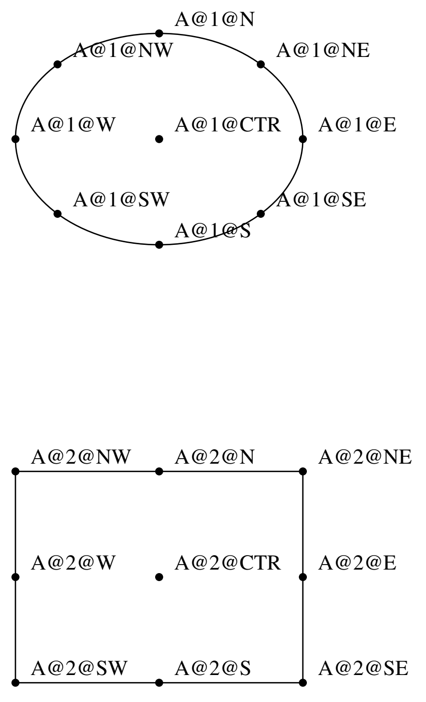
Advanced graphics

```
@SysInclude { fig }
...
@Fig {
@Box
   margin { 0c }
   paint { black }
@Ellipse
   linestyle { noline }
   paint { white }
{ Hello, world }
}
```


Hello, world

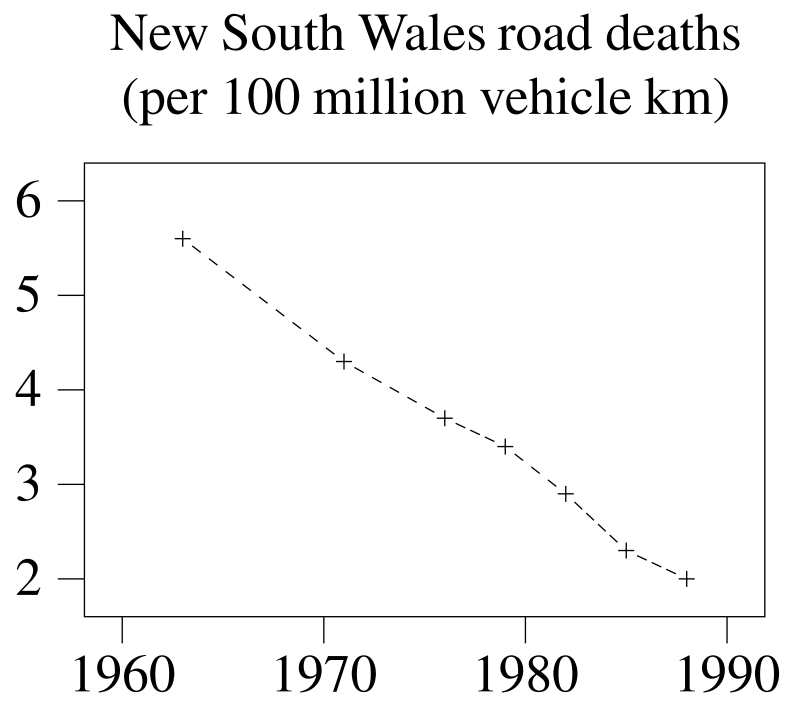
Point labelling

```
@Fig {
A::
{
    1:: @Ellipse { 3c @Wide 2c @High }
    //3c
    2:: @Box { 3c @Wide 2c @High }
}
@ShowLabels
}
```


A@1@N
A@1@NW
A@1@NE
A@1@W
A@1@CTR
A@1@E
A@1@SW
A@1@SE
A@1@S
A@2@NW
A@2@N
A@2@NE
A@2@W
A@2@CTR
A@2@E
A@2@SW
A@2@S
A@2@SE

Graphs

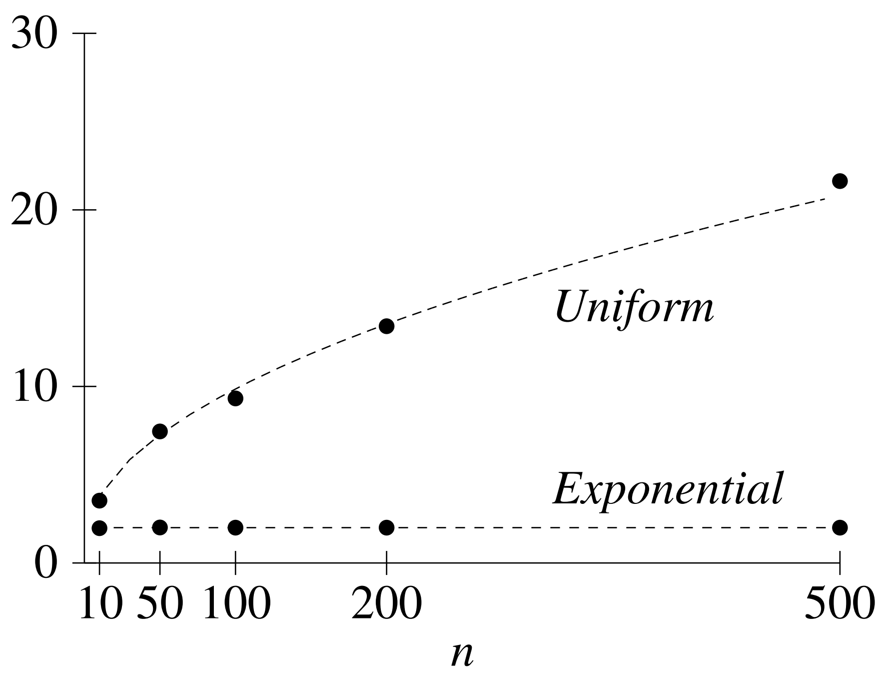
```
@Graph
   abovecaption { New South Wales road deaths
(per 100 million vehicle km) }
{
   @Data  points { plus }  pairs { dashed }
   { 1963 5.6  1971 4.3  1976 3.7  1979 3.4
   1982 2.9  1985 2.3  1988 2.0 }
}
```


New South Wales road deaths
(per 100 million vehicle km)
6
5
4
3
2
1960
1970
1980
1990

```
-2p @Font @Graph
    style { axes }
    xorigin { 0 }  yorigin { 0 }
    xticks { 10@ 50@ 100@ 200@ 500@ }
    objects {  @NE at { 300 2 } @I { Exponential }
      @SE at { ... } @I { Uniform } }
    belowcaption { @I n }
{
    @Data points { filledcircle } { ... }
    @Data points { filledcircle } { ... }

    @Data pairs { dashed }
    { 10 2  500 2 }

    @Data pairs { dashed }
    {
        xloop from { 10 } to { 500 } by { 20 } do
        {
            x  sqrt { pi*x / 4 } + 1
        }
    }
}
```


30
20
10
0
Uniform
Exponential
10 50 100 200 500
n