# More Advanced LaTeX

## How to do 'difficult' things

Christophe Rhodes

c.rhodes@gold.ac.uk

Summer skills workshop

21st July 2009

# TEX: a typesetting system

## Donald Knuth and others



TEX's development:

- Motivated by poor page proofs of "The Art of Computer Programming", vol. 2, 2nd edition
- Guy Steele, again! Lobbied (successfully) for Turing-completeness;
- Hyphenation algorithm by Frank Liang, "Word Hy-phen-a-tion by Com-put-er"
- TEX version 3 released in 1989;
  - minor updates every 4 years or so;
  - version number approaches $\pi$.
- TEX source code is effectively Public Domain.

# LaTeX: a Document Preparation System

Leslie Lamport and others



LaTeX:

- frontend to TeX;
- widely used in the academic community;
- separation of formatting and content;
- wide variety of add-ons.

Strengths:

- decent-quality output;
- reasonable future compatibility;
- good support for mathematics;
- text-based format.

# LaTeX: a Document Preparation System

## Traditional barriers



Obstacles traditionally cited as barriers to adoption:

- WYSINWYG nature;
- little low-level control;
- steep learning curve;
- non-ubiquitous availability.

And some that we'll discuss more here:

- poor support for graphics and diagrams;
- idiosyncratic font handling.

# LaTeX: a Document Preparation System

## Traditional barriers

Obstacles traditionally cited as barriers to adoption:

- WYSINWYG nature;
- little low-level control;
- steep learning curve;
- non-ubiquitous availability.

And some that we'll discuss more here:

- poor support for graphics and diagrams;
- idiosyncratic font handling.

# Outline

1. Graphics
   - basic LaTeX pictures;
   - PostScript tricks;
   - TikZ and PGF.
2. Fonts
   - what is a font anyway?
   - font packages;
   - XeTeX and OpenType fonts.
3. Classes and Packages
   - implementing Goldsmiths house style;
   - documented TeX.

# Graphics

A large number of options:

- the LaTeX picture environment;
- the graphics and graphicx packages;
  - the psfrag package;
- pstricks;
- TikZ and PGF.

# Graphics

A large number of options:

- the LaTeX picture environment;
- the graphics and graphicx packages;
  - the psfrag package;
- pstricks;
- TikZ and PGF.

# Graphics

## The picture environment



"The picture environment allows you to create just about any kind of picture you want containing text, lines, arrows and circles."

- use `\put` to put things in a picture;
- `\line`, `\vector` and `\circle` for lines, arrows and circles;

Software capable of exporting to LaTeX pictures:

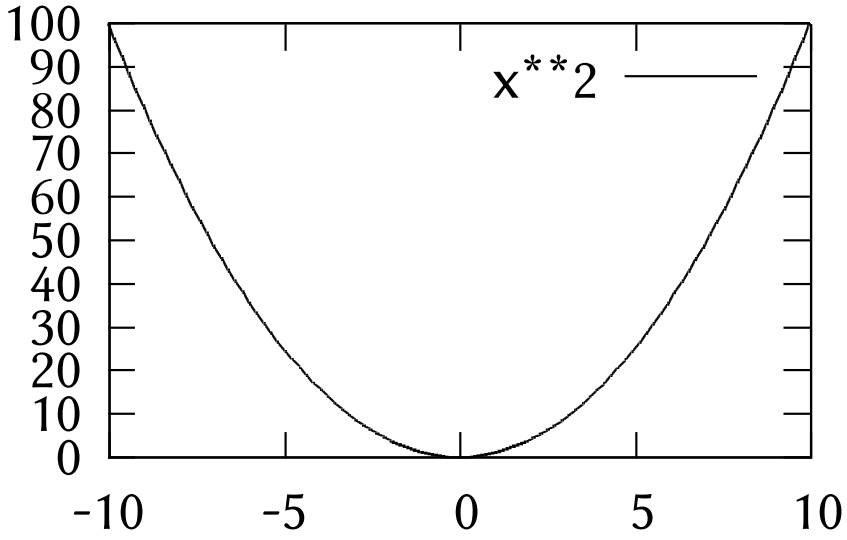
- Xfig
- Gnuplot

# Graphics

## The picture environment

Generated from Gnuplot with

- `set term latex size 2.5,1.5;`
- `plot x**2 with lines.`




Defects: poor font scaling; unconventional `x**2` label; unscalable; jaggies in the curve.

# Graphics

## The graphics and graphicx packages



Allow the inclusion of graphics in various formats:

- standard LaTeX: Encapsulated PostScript;
- pdfLaTeX and XeTeX: PDF, PNG.

Can apply scaling, image transformations and similar to the included image. For example:

```
\includegraphics[width=0.5\textwidth]{filename}
```

includes an image, scaling it so that its width is half the text width.

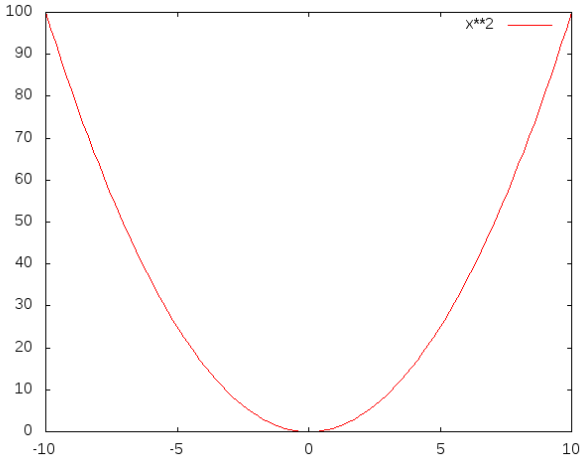
# Graphics

## The graphics and graphicx packages



Generated from Gnuplot with

- `set term png;`
- `plot x**2 with lines.`




Defects: font shape mismatch; unconventional `x**2` label; only partly scalable.

# Graphics

## psfrag and PSTricks: Unfettered PostScript



The psfrag package allows the replacement of postscript strings in an included 'image' with LaTeX-formatted fragments.

- `\psfrag{x**2}{$x^2$}`
- `\psfrag{100}{\small 100}`
- Does not work with PDF output.

The pstricks package (and many add-on packages) allow the construction of all sorts of graphics, implemented through the emission of PostScript specials:

- `\psplot x sqr` [FIXME]
- Does not work with PDF output;
  - but see the pst-pdf package.

# TikZ and PGF

PGF:

- Portable Graphics Format
- ... or "pretty, good, functional"
- the 'basic layer' of a layered graphics system for TEX

TikZ:

- TikZ ist kein Zeichenprogramm
- "TikZ is no drawing program"
- one of several 'frontend layers' for PGF

# TikZ and PGF

PGF:

- Portable Graphics Format
- ... or "pretty, good, functional"
- the 'basic layer' of a layered graphics system for TEX

TikZ:

- TikZ ist kein Zeichenprogramm
- "TikZ is no drawing program"
- one of several 'frontend layers' for PGF

# TikZ and PGF

PGF:

- Portable Graphics Format
- ... or "pretty, good, functional"
- the 'basic layer' of a layered graphics system for TEX

TikZ:

- TikZ ist kein Zeichenprogramm
- "TikZ is no drawing program"
- one of several 'frontend layers' for PGF

# TikZ and PGF

## The `tikzpicture` environment



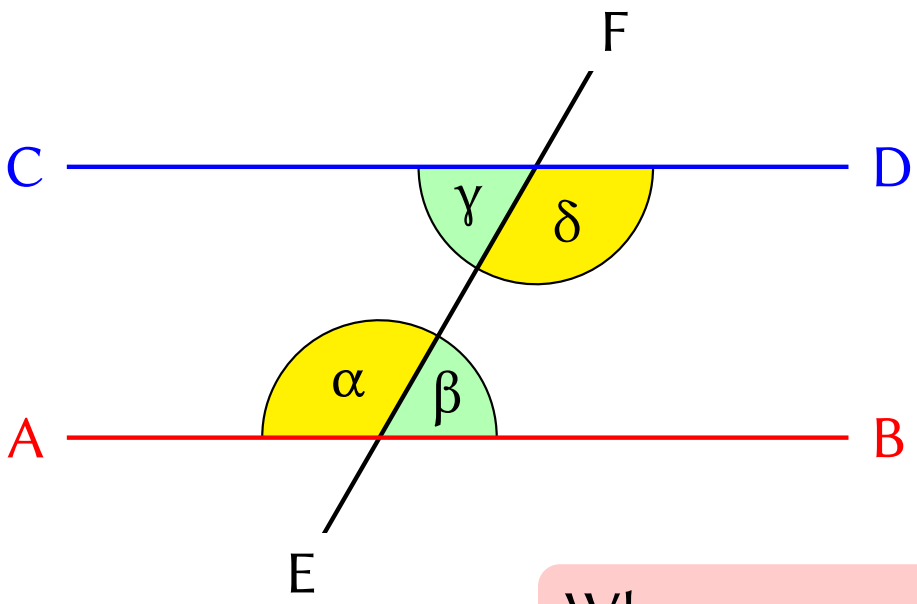
The `tikzpicture` environment surrounds the commands to construct a picture or diagram.
Shortcuts:

- `\tikz[options]{commands}`
- `\tikz[options] text;`

Within the `tikzpicutre` environment go commands to construct, stroke and fill paths, similar to many graphical systems in common use.

# TikZ and PGF

## A tikz demonstration

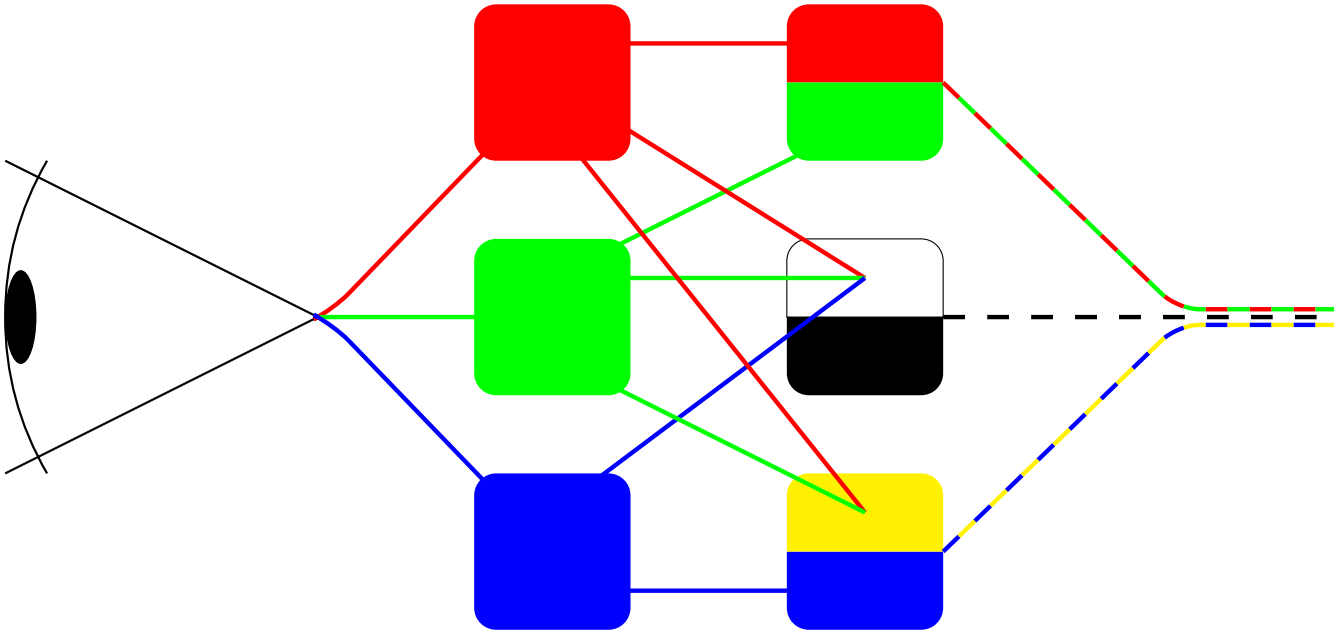



When we assume that AB and CD are parallel, i.e., AB∥CD, then $\alpha=\delta$ and $\beta=\gamma$.

(Lightly modified from the TikZ and PGF manual)

# TikZ and PGF

## A tikz demonstration



The opponent model of visual perception, from CC227 slides

# TikZ and PGF

## Simple paths

Straight lines:

- `\draw (0,0)--(2,0);`
- `\draw (0,0)--(1,0.25)--(2,0);`
- `\draw (0,0)--(0.5,0)--(0.5,0.5)--(0,0.5)--cycle;`

Bézier curves:

- `\draw (0,0)..controls (0.4,0.2) and (0.6,0.3)..(1,0);`

Circles, ellipses and arcs:

- `\draw (0,0) circle (0.25);`
- `\draw (0,0) ellipse (0.4 and 0.2);`
- `\draw (0,0) arc (270:45:0.2);`

# TikZ and PGF

## Simple paths



### Straight lines:

- `\draw (0,0)--(2,0);`
- `\draw (0,0)--(1,0.25)--(2,0);`
- `\draw (0,0)--(0.5,0)--(0.5,0.5)--(0,0.5)--cycle;`

### Bézier curves:

- `\draw (0,0)..controls (0.4,0.2) and (0.6,0.3)..(1,0);`

### Circles, ellipses and arcs:

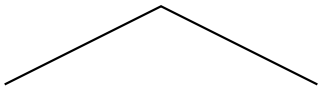
- `\draw (0,0) circle (0.25);`
- `\draw (0,0) ellipse (0.4 and 0.2);`
- `\draw (0,0) arc (270:45:0.2);`

# TikZ and PGF

## Simple paths



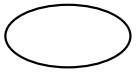
### Straight lines:

- `\draw (0,0)--(2,0);` 
- `\draw (0,0)--(1,0.25)--(2,0);` 
- `\draw (0,0)--(0.5,0)--(0.5,0.5)--(0,0.5)--cycle;` 

### Bézier curves:

- `\draw (0,0)..controls (0.4,0.2) and (0.6,0.3)..(1,0);` 

### Circles, ellipses and arcs:

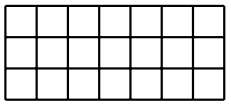
- `\draw (0,0) circle (0.25);` 
- `\draw (0,0) ellipse (0.4 and 0.2);` 
- `\draw (0,0) arc (270:45:0.2);` 

# TikZ and PGF

## More esoteric paths

Grid:

- `\draw (0,0) grid[step=0.2] (1.4,0.6);`



Sine and cosine:

- `\draw (0,0)sin(0.5,1)cos(1,0)sin(1.5,-1)cos(2,0);`



Plot:

- `\draw plot(\x,0.2*\x);`



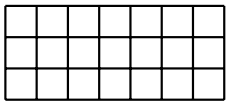
- `\draw plot(\x,{0.2*sin(3*\x r)});`

# TikZ and PGF

## More esoteric paths

Grid:

- `\draw (0,0) grid[step=0.2] (1.4,0.6);`



Sine and cosine:

- `\draw (0,0)sin(0.5,1)cos(1,0)sin(1.5,-1)cos(2,0);`

Plot:

- `\draw plot(\x,0.2*\x);`

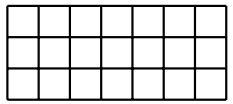


- `\draw plot(\x,{0.2*sin(3*\x r)});`

# TikZ and PGF

## More esoteric paths

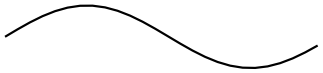
Grid:

- `\draw (0,0) grid[step=0.2] (1.4,0.6);` 

Sine and cosine:

- `\draw (0,0)sin(0.5,1)cos(1,0)sin(1.5,-1)cos(2,0);` 

Plot:

- `\draw plot(\x,0.2*\x);` 
- `\draw plot(\x,{0.2*sin(3*\x r)});` 

Nodes are text drawn on top of a path:

- `\draw (0,0) node {foo} -- (1,0) node {bar};` foo bar
- `\draw (0,0) node[draw] {foo} -- (1,0) node[draw] {bar};` foo bar

Nodes define anchors:

```
\node[draw] (foo) at (0,0) {foo};
\node[draw] (bar) at (1,0) {bar};
\draw(foo.east) -- (bar.west);
```

foo bar

Many shapes of node are implemented:

- rectangle, circle, and other geometric shapes;
- forbidden sign, cloud, starburst, signal, tape;
- single arrow, double arrow, arrow box;
- logic gates, and more!

Nodes are text drawn on top of a path:

- `\draw (0,0) node {foo} -- (1,0) node {bar};` foo bar
- `\draw (0,0) node[draw] {foo} -- (1,0) node[draw] {bar};` foo bar

Nodes define anchors:

```
\node[draw] (foo) at (0,0) {foo};
\node[draw] (bar) at (1,0) {bar};
\draw(foo.east) -- (bar.west);
```

foo bar

Many shapes of node are implemented:

- rectangle, circle, and other geometric shapes;
- forbidden sign, cloud, starburst, signal, tape;
- single arrow, double arrow, arrow box;
- logic gates, and more!

Nodes are text drawn on top of a path:

- `\draw (0,0) node {foo} -- (1,0) node {bar};` foo bar
- `\draw (0,0) node[draw] {foo} -- (1,0) node[draw] {bar};` foo bar

Nodes define anchors:

```
\node[draw] (foo) at (0,0) {foo};
\node[draw] (bar) at (1,0) {bar};
\draw(foo.east) -- (bar.west);
```

foo bar

Many shapes of node are implemented:

- rectangle, circle, and other geometric shapes;
- forbidden sign, cloud, starburst, signal, tape;
- single arrow, double arrow, arrow box;
- logic gates, and more!

# TEX and METAFONT

- path-and-stroke description of glyphs;
- based on geometric descriptions and cubic splines;
- parametrized fonts (hence 'META'FONT);

path-and-stroke model

# TEX and METAFONT

## The Computer Modern family



The Computer Modern family is almost synonymous with the LATEX 'look'

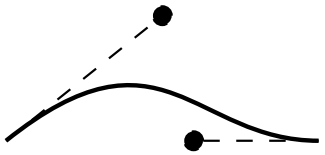
- Computer Modern Roman
- *Computer Modern Oblique*
- *Computer Modern Italic*
- **Computer Modern Bold**
- ***Computer Modern Bold Italic***
- COMPUTER MODERN CAPS
- Computer Modern Sans Serif
- `Computer Modern Typewriter`

Accompanying mathematics fonts also come from the same METAFONT definitions

# PostScript fonts

Type 1 PostScript fonts:

- outline descriptions of glyphs;
- based on cubic Bézier curves;



- tables for ‘hinting’.

Historically used in printers, and in the X Window System.

# PostScript fonts

## Charter and Utopia



PostScript Type 1 fonts (usually) distributed with LaTeX:

- Adobe Utopia;
- Bitstream Charter;

Usable by default in LaTeX with

- `\usepackage{utopia}` and
- `\usepackage{charter}`

respectively

# PostScript Fonts

## Other PostScript Fonts



Fonts from URW++ Design & Development:

- URW Antiqua 2051 Regular Condensed;
- URW Avantgarde;
- **URW Bookman;**
- **URW Grotesque;**
- URW Palatino;
- *URW Zapf Chancery Italic.*

Times Roman, Helvetica and Courier:

- set of three (serif, sans and monospace) in a single document;
- typically installed on printers;
- `\usepackage{times}`

# PostScript Fonts

## Other PostScript Fonts



Fonts from URW++ Design & Development:

- URW Antiqua 2051 Regular Condensed;
- URW Avantgarde;
- **URW Bookman;**
- **URW Grotesque;**
- URW Palatino;
- *URW Zapf Chancery Italic.*

Times Roman, Helvetica and Courier:

- set of three (serif, sans and monospace) in a single document;
- typically installed on printers;
- `\usepackage{times}`

# TrueType fonts

TrueType fonts:

- outline descriptions of glyphs;
- based on quadratic Bézier curves;

- (patented) algorithms for hinting;

Historically developed by Apple and licenced for use in the Microsoft Windows Operating System

# OpenType fonts

OpenType fonts:

- outline descriptions of glyphs;
- wrap either PostScript or TrueType glyph data;
- include metadata regarding scripts;

Microsoft and Adobe collaboration, intended to supersede both Type 1 and TrueType fonts: ISO/IEC 14496-22

# OpenType fonts

## XƎTEX and the fontspec package



XƎTEX is a modified TEX engine:

- uses 'words' rather than characters as the basic building block
  - breaking words up if it is necessary to hyphenate
- allows transparent use of system-installed fonts
  - TrueType and OpenType by default;
  - Also allows PostScript fonts to be used if specified
- accepts UTF-8 input;
- outputs to PDF.

# OpenType fonts

## XeTeX and the fontspec package

The fontspec package:

- works with the XeTeX version of LaTeX;
- easy access to fonts;
- straightforward access to those fonts' features:
  - alternate forms;
  - rare ligatures and swashes;
  - variant numeral styles;
  - kerning, tracking and optical justification;

# Classes and Styles

Style files:

- used with `\usepackage`;
- provide added functionality
  - or adapt basic functionality.
- a document can have any number of styles loaded;
  - some are mutually exclusive;
  - a very few are mutually incompatible.
- examples: `url`, `hyperref`, `ifthen`, `amsmath`, `geometry`

Class files:

- used with `\documentclass`;
- provide the basic functionality and layout for a document;
- a document has exactly one class loaded
  - though that class can reuse a more basic class in its implementation with `\LoadClass`
- examples: `article`, `report`, `book`, `beamer`, `letter`

# Classes and Styles

Style files:

- used with `\usepackage`;
- provide added functionality
  - or adapt basic functionality.
- a document can have any number of styles loaded;
  - some are mutually exclusive;
  - a very few are mutually incompatible.
- examples: `url`, `hyperref`, `ifthen`, `amsmath`, `geometry`

Class files:

- used with `\documentclass`;
- provide the basic functionality and layout for a document;
- a document has exactly one class loaded
  - though that class can reuse a more basic class in its implementation with `\LoadClass`
- examples: `article`, `report`, `book`, `beamer`, `letter`

# Classes and Styles

## The minimal style file



A minimal style file contains two things:

1. a declaration of the minimum LaTeX requirements it needs, using the `\NeedsTeXFormat` macro;
2. a declaration of what the style file provides, using the `\ProvidesPackage` macro.

Example:

```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesPackage{foo}
  [2009/07/21 v0.1 Goldsmiths ‘foo’ package]
```

# Classes and Styles

## Less minimal style files



Normally, style files contain more functionality than that. They can

- load other style files, using `\RequirePackage`
- define LaTeXmacros, using `\newcommand` or `\renewcommand`
  - or the lower-level TeX defining macros `\def` or `\edef`
- define LaTeXenvironments, using `\newenvironment` or `\renewenvironment`
- declare pieces of code to be run when options are passed to the package, using `\DeclareOption`
  - and using `\ProcessOptions` to actually run the code.

# Classes and Styles

## Minimal Class file



A minimal class file contains the same information as a minimal style file:

- a `\NeedsTeXFormat` declaration;
- a `\ProvidesClass` declaration;

but must also provide four definitions:

1. `\textwidth`, the width of the typeset text on the page;
2. `\textheight`, the height of the typeset text on the page;
3. `\normalsize`, a declaration for the size of the normal font;
4. a page numbering specification.

# Classes and Styles

## Minimal Class file



```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\ProvidesClass{foo}

% This is not 100% minimal, in that size11.clo
% defines not just \textheight, \textwidth (based on
% \paperheight and \paperwidth) and \normalsize, but
% also a number of other things.
\setlength{\paperheight}{297mm}
\setlength{\paperwidth}{210mm}
\input{size11.clo}

% In modern \LaTeX distributions this isn't in fact
% necessary, as it comes predefined in the kernel.
\pagenumbering{arabic}
```

# Documented TEX

The Documented TEX conventions are a set of rules, and some tools, for writing and documenting TEX and LATEX software.

- The docstrip package reads docmented TEX and produces class files, style files, and any other ancillary files required to use the software
- The doc package reads documented TEX and produces documentation, which can include
  - a user guide;
  - a change log;
  - a listing of the software;
  - a cross-referenced index of all macros described, used or defined.

# Documented TEX

## Aside: Literate Programming



Literate Programming is a style of programming which encourages or forces the author of the code to explain the thought process behind it. Potential benefits:

- reveals design errors;
- easier to keep documentation up to date with code;
- gives reward feedback for writing documentation
  - increases likelihood of documentation being written in the first place;

As well as documented TEX, systems for literate programming include

- web, including cweb and noweb
- pbook.el

# Documented TEX

## Aside: Literate Programming



Literate Programming is a style of programming which encourages or forces the author of the code to explain the thought process behind it. Potential benefits:

- reveals design errors;
- easier to keep documentation up to date with code;
- gives reward feedback for writing documentation
  - increases likelihood of documentation being written in the first place;

As well as documented TEX, systems for literate programming include

- web, including cweb and noweb
- pbook.el

# Documented TEX

## The structure of a documented TEX file



The structure of a documented TEX file is complicated:

- written in several 'languages' simultaneously;
- can produce multiple output files;
- comments, meta-comments and directives.

I don't have a marvellous description, but even if I did this slide would be too short to contain it:

- work by example.