

# TEX: Rejoining the mainstream

Jonathan Fine

EuroTeX 2009

The Hague, Netherlands

1 September 2009

Slides at [mathtran docs on SVN on Sourceforge](#)

## Jon Udell's insight on T<sub>E</sub>X, L<sup>A</sup>T<sub>E</sub>X and the web

In 2000 Jon Udell, in a report on *Internet Groupware for Scientific Collaboration* wrote:

*T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X define scientific publishing for a generation of scientists. But these formats don't integrate directly into the shared spaces of the Web.*

In July 2009, after encountering the Polymath project, he wrote:

*Why didn't I see, then, that the crux of the issue wasn't XML and MathML and SVG, but rather the ability to **"integrate directly into the shared spaces of the Web"**? And that what ought to be integrated directly was the typesetting language already familiar to mathematicians, namely L<sup>A</sup>T<sub>E</sub>X?*

<http://blog.jonudell.net/2009/07/31/polymath-equals-user-innovatio/>

## Knuth's vision

*[T]he T<sub>E</sub>X research project ... was driven by two major goals ... **quality** ... the best ... **archival** ... 100 years*

*I'm not going to design a programming language; I want to have just a typesetting language.*

*In some sense I put in many of T<sub>E</sub>X's programming features only after kicking and screaming [from users].*

*Now, if there were a universal simple interpretive language that was common to other systems, naturally I would have latched onto that right away.*

*Let us regard these systems [T<sub>E</sub>X and Metafont] as fixed points, which should give the same results 100 years from now that they produce today.*

Quotes from **Digital Typography**, pages 559, 648, 648, 649, 571.

## Knuth's request: use another name

When he announced, in 1990, that his work on developing T<sub>E</sub>X, Metafont and Computer Modern had come to an end, (except for extremely serious bugfixes), Don Knuth wrote (DT, p571):

*I welcome continued research that will lead to alternative systems that can typeset documents better than T<sub>E</sub>X is able to do. But the authors of such systems must think of another name.*

*That is all I ask, after devoting a substantial portion of my life to the creation of these systems. I sincerely hope that the members of TUG will help me to enforce these wishes, by putting severe pressure on any person or group who produces an incompatible system and calls it T<sub>E</sub>X or METAFONT or Computer Modern — no matter how slight the incompatibility might seem.*

# Introduction and overview

This talk is a simplified history of T<sub>E</sub>X: past, present and future. Mainstream was print, now includes Web. The basic ideas are:

- ▶ T<sub>E</sub>X was mainstream end-user software.
- ▶ T<sub>E</sub>X is no longer in mainstream of open source.
- ▶ T<sub>E</sub>X can rejoin the mainstream.
- ▶ Helping T<sub>E</sub>X rejoin the mainstream.

By the word 'T<sub>E</sub>X' we mean one of:

- ▶ The program `tex` written by Don Knuth.
- ▶ A backslash and braces markup language.
- ▶ Software and other resources associated with T<sub>E</sub>X.
- ▶ A community of users and developers.

But not (as requested by DEK) some other typesetting program.

## Achievements of $\text{\LaTeX}$ (the first fifteen years)

- ▶ Provides styles (aka classes) for standard documents
- ▶ Open: Allows programmers to add their own style files
- ▶ Provides reasonably uniform document syntax
- ▶ Lamport's  $\text{\LaTeX}$  book provides gentle introduction
- ▶ Attracted energies of many talented developers
- ▶ Used and respected by many academics
- ▶ Adopted as submission format by many publishers
- ▶  $\text{\LaTeX}2e$  an important clean-up and revision of  $\text{\LaTeX}209$

# L<sup>A</sup>T<sub>E</sub>X and Python documentation compared

Please visit <http://www.latex-project.org/guides/>

- ▶ First link on page is to books (mostly by team members).
- ▶ Second link is to 33 page PDF [2001, stale addresses].
- ▶ Most of the other links are to 3rd party PDF files.
- ▶ HTML links to wikibooks and Andy Roberts sites.
- ▶ Doesn't link to Indian TUG Tutorial, Nicola Talbot, ...

Please visit <http://docs.python.org/>

- ▶ Docs available in HTML and PDF
- ▶ Search page for docs
- ▶ Online Global Module Index and General Index
- ▶ Permalinks to page fragments
- ▶ Syntax highlighted code
- ▶ Copyright, the Python Software Foundation



# LaTeX documentation

LaTeX – A document preparation system

[LaTeX project site](#) → Documentation

[LaTeX home](#)

[Introduction](#)

[LaTeX news](#)

[Documentation](#)

[Books](#)

[LaTeX3 project](#)

[Project articles](#)

[Development code](#)

[LaTeX3 news](#)

[How to get it](#)

[Get help!](#)

[Browse bugs database](#)

[How to report a bug](#)

[Site news](#)

[Site news feed](#)

[The LPP!](#)

[Contact](#)

Search

## News

[«The LaTeX graphics companion, 2nd edition»](#)

This page contains references to documentation that is available on the net. Most of these are in English but there is a section on non-English documentation. They are by both the LaTeX team and other contributors. Also, there's a whole page dedicated to [books on LaTeX and related topics](#).

If you know about a list of TeX and related documentation not listed here, please contact [webmaster@latex-project.org](mailto:webmaster@latex-project.org).

## Documentation distributed with LaTeX

LaTeX is described in *LaTeX2e for authors* which is available as a [LaTeX document](#) or as a [PDF document](#).

More advanced documentation about LaTeX, which is available via the net includes:

- [Configuration options for LaTeX2e](#)
- [LaTeX2e for class and package writers](#)
- [Cyrillic languages support in LaTeX](#)
- [LaTeX2e font selection](#)
- [The LaTeX3 project](#)
- [Modifying LaTeX](#)

## Contributed documentation



## Download

Download these documents

## Docs for other versions

[Python 2.7 \(in development\)](#)  
[Python 3.1 \(stable\)](#)  
[Python 3.2 \(in development\)](#)  
[Old versions](#)

## Other resources

[FAQs](#)  
[Guido's Essays](#)  
[New-style Classes](#)  
[PEP Index](#)  
[Beginner's Guide](#)  
[Book List](#)  
[Audio/Visual Talks](#)  
[Other Doc Collections](#)

## Quick search

Go

Enter search terms or a module, class or function name.

# Python v2.6.2 documentation

Welcome! This is the documentation for Python 2.6.2, last updated Aug 31, 2009.

## Parts of the documentation:

### What's new in Python 2.6?

*or all "What's new" documents since 2.0*

### Tutorial

*start here*

### Using Python

*how to use Python on different platforms*

### Library Reference

*keep this under your pillow*

### Language Reference

*describes syntax and language elements*

### Python HOWTOs

*in-depth documents on specific topics*

### Extending and Embedding

*tutorial for C/C++ programmers*

### Python/C API

*reference for C/C++ programmers*

### Installing Python Modules

*information for installers & sys-admins*

### Distributing Python Modules

*sharing modules with others*

### Documenting Python

*guide for documentation authors*

## Indices and tables:

### Global Module Index

*quick access to all modules*

### General Index

### Search page

*search this documentation*

## L<sup>A</sup>T<sub>E</sub>X3 project — digging deeper

- ▶ Started in 1993 or so (predates XML, Google, ...)
- ▶ No-one is using L<sup>A</sup>T<sub>E</sub>X3 for typesetting
- ▶ In 2005, L<sup>A</sup>T<sub>E</sub>X3 source placed on SVN server, but ...
- ▶ They say it's **explicitly forbidden** to publish L<sup>A</sup>T<sub>E</sub>X3 code
- ▶ Uses unusual license (Debian accepted, not OSI-approved)
- ▶ Current activity focused new macro programming interface

Here's an example of the old and new interface:

```
\def\mymacro #1{\setbox #1\hbox\bgroup} % Old
\cs_new_nopar:Npn \hbox_set_inline_begin:N #1 { % New
  \tex_setbox:D #1 \tex_hbox:D \c_group_begin_token }
```

It doesn't even get named parameters (instead of #1).

## Rebuttal from Will Robertson ( $\LaTeX$ 3 team member)

[The new syntax] isn't just about renaming TeX primitives for the sake of it:

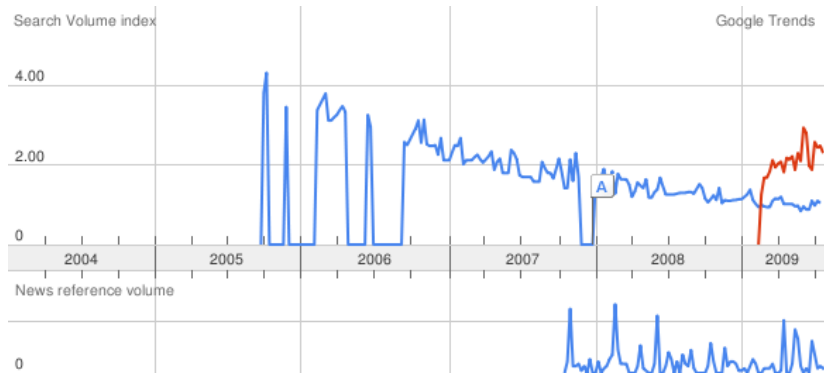
- ▶ toolbox of often-used and otherwise useful functions with consistent (and readable!) names
- ▶ abstract many expansion control problems with better datatypes

[Here's an example]: (from LaTeX2e kernel)

```
\global\expandafter\let                                %% Old
  \csname\cf@encoding \string#1%
    \expandafter\endcsname
    \csname ?\string#1\endcsname
\cs_gset_eq:cc                                        %% New
  { \cf@encoding \token_to_str:N #1 }
  { ? \token_to_str:N #1 }
```

# Google trends for MathML

<http://www.google.com/trends?q=MathML,+lmgfny>



MathML in blue (from 2005), lmgfny in red (from 2009).

## Terry Tao's blog and the Polymath project

Terry Tao is a young and extremely eminent mathematician. He's an early adopter. His blog <http://terrytao.wordpress.com> is published in book form. Typeset in  $\text{\LaTeX}$ , of course.

Along with Tim Gowers, Gil Kalai and Michael Nielsen, he's set up <http://polymathprojects.org>. (And see also <http://www.tricki.org>)

Polymath is all interested people across the world working together on a problem of common interest. It's using a WordPress blog (with  $\text{\LaTeX}$  support).

Computer industry commentator Jon Udell wrote of Polymath:

*Why didn't I see, then [2000], that the crux of the issue wasn't XML and MathML and SVG, but rather the ability to "integrate directly into the shared spaces of the Web"? And that what ought to be integrated directly was the typesetting language already familiar to mathematicians, namely  $\text{\LaTeX}$ ?*

# The polymath blog



August 9, 2009

## (Research thread II) Deterministic way to find primes

Filed under: [finding primes](#), [research](#) — Terence Tao @ 3:57 am  
Tags: [polymath4](#)

This thread marks the formal launch of “[Finding primes](#)” as the massively collaborative research project Polymath4, and now supersedes the [proposal thread for this project](#) as the official “research” thread for this project, which has now become rather lengthy. (Simultaneously with this research thread, we also have the [discussion thread](#) to oversee the research thread and to provide a forum for casual participants, and also the [wiki page](#) to store all the settled knowledge and accumulated insights gained from the project to date.) See also this [list of general polymath rules](#).








The basic problem we are studying here can be stated in a number of equivalent forms:

**Problem 1.** (Finding primes) Find a *deterministic* algorithm which, when given an integer  $k$ , is *guaranteed* to locate a prime of at least  $k$  digits in length in as quick a time as possible (ideally, in time polynomial in  $k$ , i.e. after  $O(k^{O(1)})$  steps).

**Problem 2.** (Finding primes, alternate version) Find a *deterministic* algorithm which, after running for  $k$  steps, is *guaranteed* to locate as large a prime as possible (ideally, with a polynomial number of digits, i.e. at least  $k^c$  digits for some  $c > 0$ ).

To make the problem easier, we will assume the existence of a *primality oracle*, which can test whether any given number is prime in  $O(1)$  time, as well as a *factoring oracle*, which will provide all the factors of a given number in  $O(1)$  time. (Note that the latter supersedes the former.) The primality oracle can be provided essentially for free, due to polynomial-time deterministic primality algorithms such as the [AKS primality test](#); the factoring oracle is somewhat more expensive (there are deterministic factoring algorithms, such as the [quadratic sieve](#), which are suspected to be subexponential in running time, but no polynomial-time algorithm is known), but seems to simplify the problem substantially.

### Recent Comments

-  [Mark Lewko on \(Research Thread IV\) Deterministic way to find primes...](#)
-  [Mark Lewko on \(Research Thread IV\) Deterministic way to find primes...](#)
-  [H A Helfgott on \(Research Thread IV\) Deterministic way to find primes...](#)
-  [Mark Lewko on \(Research Thread IV\) Deterministic way to find primes...](#)
-  [Gil Kalai on \(Research Thread IV\) Deterministic way to find primes...](#)
-  [Mark Lewko on Deterministic way to find primes...](#)
-  [Mark Lewko on Deterministic way to find primes...](#)

approach. What we did was to look at an interval of length  $k^2$  of  $k$ -digit numbers,  $[k^m, k^m + k^2]$ , and to conjecture that some number in the interval has a large prime factor. (Where "large" can be a number with at least  $m$  digits for  $m = k/\log k$  or even just  $m = k^{1/3}$  numbers.)

A weaker conjecture that will be as fine for us is this: If we start with an interval of length  $k^5$  and consider the set  $S$  of all  $m$ -digit or more factors (not necessarily primes) of all the numbers in the intervals; then there will be an  $m$ -digit prime factor for a number in  $S+S$ .

In other words, we can allow several iterations of the operations: "taking sums" and "taking factors".

The razor is as bad to this idea as it was for the original one. But if we can somehow find a way around the razor, this extension can be helpful because the sum/product theorems suggest that you can do more if you allow more than one arithmetic operation.

✓ 0 ✗ 0 Rate This

Comment by *Gil* — August 14, 2009 @ 6:14 am | Reply

49. [...] finding primes, research — Terence Tao @ 5:10 pm Tags: polymath4 This is a continuation of Research Thread II of the "Finding primes" polymath project, which is now full. It seems that we are [...]



✓ 0 ✗ 0 Rate This

Pingback by [\(Research Thread III\) Deterministic way to find primes « The polymath blog](#) — August 13, 2009 @ 5:10 pm | Reply

50. As this thread is now quite full, I am opening a new research thread for this project at



<http://polymathprojects.org/2009/08/13/research-thread-iii-deterministic-way-to-find-primes/>

✓ 0 ✗ 0 Rate This

Comment by *Terence Tao* — August 13, 2009 @ 5:12 pm | Reply

Used in production by the SIL linguistics institute.

An open source typesetting program based on a merger of Donald Knuth's T<sub>E</sub>X system with Unicode and modern (OpenType and AAT) font technologies.

The XeTeX typesetting system is cross-platform. It provides the same Unicode and OpenType font support on GNU/Linux, Mac OS X and Windows (but no AAT font support outside Mac OS X).

Developer Jonathan Kew now works for Mozilla, on internationalisation.



Recall Don Knuth's goal was to create *just a typesetting language*.  
LuaTeX is going in the opposite direction.

- ▶ It *embeds* Lua into an extension of TeX.
- ▶ Describes itself as a 'variant of TeX' (loud tutting).
- ▶ For many (most) users, not better than XeTeX.
- ▶ Embed rather than extend is a *big mistake*.
- ▶ Gives Lua access to typesetting internals.
- ▶ Big performance and memory hit.
- ▶ Requires extensive rewrite of Don Knuth's code for TeX.

In my view, XeTeX provides now all the benefits LuaTeX will provide, except: Lua is better than the TeX macro language.  
However, there are other ways to obtain this benefit.

## Embed and extend compared: client and server

We *extend* a client to connect to a server. For example, in Python to connect to an SQL database we import an extension module.

We can also *embed* Python in an SQL database server. This allows SQL queries to use Python, for example to filter a result set.

With databases, almost always we extend (with an extension module). It's rare to use Python embedded in a database server. Apache is the only common use I know of Python embedded (via `mod_python`).

It's common to import into Python many extension modules. But there's no easy way to combine two embeddings.

Lua $\TeX$  chooses embed because  $\TeX$  has an embedded macro language (which was probably right then). The key thing now is to provide a service, not a better embedded language.

# Transform, typeset, make-up

In T<sub>E</sub>X, three distinct processes are combined.

- ▶ Transform input document into horizontal list
- ▶ Breaking the paragraph into lines
- ▶ Combining the paragraphs into pages

The same goes for L<sup>A</sup>T<sub>E</sub>X, ConT<sub>E</sub>Xt and LuaT<sub>E</sub>X.

**Transform** is all about fonts, sizes, colours and generated text (such as cross-references).

**Breaking** is a key component of any typesetting system.

**Combining** is deciding what should go on what should go on each page (lines, figures, floats) and putting them together nicely).

**Transform** and **combining** can be done by an external program that does not know about typesetting.

**Breaking** can be done by an extension of T<sub>E</sub>X that outputs a stream of boxes (rather than a file of dvi).

## MathTran: Flickr for formulas

Wikipedia uses  $\LaTeX$  for complex math formulas, and provides a nice page of input/output examples. But we can do better . . .

[http://en.wikipedia.org/wiki/Help:Displaying\\_a\\_formula](http://en.wikipedia.org/wiki/Help:Displaying_a_formula)

Service is good. MathTran provides  $\TeX$  typesetting (and production of images) as a public web service.

<http://www.mathtran.org/>
















- ▶ Funded by JISC and the Open University.
- ▶ Developed mostly by speaker.
- ▶ Provide instant preview editor for formulas.
- ▶ Runs  $\TeX$  as a daemon.
- ▶ Create images that are too cheap to cache.
- ▶ Now uses Pinax/Django to provide social website.
- ▶ Requires interactive online documentation.
- ▶ Hope soon to allow authoring of such documentation.

## Larger expressions

[\[edit\]](#)

### Subscripts, superscripts, integrals

[\[edit\]](#)

Feature	Syntax	How it looks rendered	
		HTML	PNG
Superscript	<code>a^2</code>	$a^2$	
Subscript	<code>a_2</code>	$a_2$	
Grouping	<code>a^{2+2}</code>	$a^{2+2}$	
	<code>a_{i,j}</code>	$a_{i,j}$	
Combining sub & super without and with horizontal separation	<code>x_2^3</code>	$x_2^3$	
	<code>{x_2}^3</code>	$x_2^3$	
Super super	<code>10^{10^{8}}</code>	$10^{10^8}$	
Super super	<code>10^{10^{\overset{8}{}}}}</code>	$10^{10^8}$	
Super super (wrong in HTML in some browsers)	<code>10^{10^8}</code>	$10^{10^8}$	
Preceding and/or additional sub & super	<code>\sideset{1^2}{3^4}\prod_a^b</code>	$2 \prod_1^b \prod_3^4 a$	
	<code>{1}^2\!\!\Omega_3^4</code>	$2 \Omega_3^4$	
Stacking	<code>\overset{\alpha}{\omega}</code>	$\overset{\alpha}{\omega}$	
	<code>\underset{\alpha}{\omega}</code>	$\underset{\alpha}{\omega}$	
	<code>\overset{\alpha}{\underset{\gamma}{\omega}}</code>	$\overset{\alpha}{\underset{\gamma}{\omega}}$	
	<code>\stackrel{\alpha}{\omega}</code>	$\stackrel{\alpha}{\omega}$	

### Add a new formula

Title

Assignment 3

Preview

$$\Psi_m(x, t) =$$

Source

`\Psi_m(x, t) =`

Description

Is public

*If public, can be seen by any visitor to the site.*

Tags

Add

### Help area

sm358

1 2 3 4

- Schrodinger's equation (1)
- Schrodinger's equation (2)
- Stationary state
- Time-independent Schrodinger equation (1)
- Time-independent Schrodinger equation (2)
- Transmission coefficient, step
- Uncertainty
- Uncertainty (principle)
- Wave packet

### Stationary state

$$\Psi_n(x, t) = \psi_n(x) e^{-iE_n t/\hbar}$$

$$\begin{aligned} \backslash\Psi_n(x, t) \\ = \backslash\psi_n(x) \{ \backslash\rm e \}^{\{- \backslash\rm i \} E_n t / \backslash\hbar} \end{aligned}$$

Note `{\rm e}` and `{\rm i}` are used to produce Roman e and i. `\hbar` is used for Planck's constant/2pi.

## Python documents: latex2html and Sphinx

Python documentation was produced using customized latex2html (a Perl script). Provides both HTML and PDF output.  $\LaTeX$  source documents.

Since Python 2.6, documentation authored in *Restructured Text* and translated by Sphinx into HTML, Windows HTML Help, and  $\LaTeX$  for PDF. See <http://sphinx.pocoo.org/>

Sphinx says *many of its strengths come from the power and straightforwardness of reStructuredText and its parsing and translating suite, the Docutils.*

Python did not adopt plas $\TeX$ , a successor to latex2html which implements  $\TeX$ 's markup language, in Python.

Sphinx and plas $\TeX$  are pointers to  $\LaTeX$  succession.

# Conclusions

The underlying theme is to more easily *integrate directly into the shared spaces of the Web* (Jon Udell, 2009).

- ▶ Interactive online  $\LaTeX$  documentation.
- ▶ Tools for authoring and displaying mathematics on the web.
- ▶ Copy-and-paste (standards for  $\TeX$ -encoded math).
- ▶ Evolve  $\LaTeX$  into standard wiki-like markup language.
- ▶ Provide web services.
- ▶ Print backend for web pages (a *big win*).

A big problem in achieving this is legacy: documents, macros and ways of working. So we'll also need

- ▶ Regression tests for documents.
- ▶ Enthusiastic and committed developers.
- ▶ Engagement with young people.



## What next: Online $\text{\LaTeX}$ documentation

A Google search for latex+quote produces Sheldon Green's hypertext help from 1995 as top result.

Con $\text{\TeX}$ t has <http://wiki.contextgarden.net/> — much better than the  $\text{\LaTeX}$  offering. It also allow try out Con $\text{\TeX}$ t without installing it. This is a step in the right direction.

We need interactive and social help for  $\text{\TeX}$  and  $\text{\LaTeX}$ .

## What next: Firefox Mathematics plugin

The idea here is to provide a toolbar button, or some other interface, that allows you to add, view and edit mathematics on a web page.

Mozilla's Ubiquity is designed to make it easy to do such things.

<http://labs.mozilla.com/projects/ubiquity/>

Key developer for Ubiquity is Aza Raskin, head of user experience at Mozilla Labs.

Aza Raskin was previously with Humanized, who created a Windows 'desktop assistant' Enso (similar, for the aged, to Borland Sidekick).

Enso provided a  $T_{E}X$  *anywhere* component that uses MathTran's web service for typesetting (and has a neat *unrender* function).

# What next: Google Summer of Code

Google pays about 1,000 students to write open source software during the summer.

**2008** Accepted for first time as mentoring organisation. Three projects (one mentored by myself).

None produced code that is in use.

**2009** Not accepted as mentoring organisation.

However, AbiWord, Django, Drupal, Inkscape, MoinMoin Wiki, Moodle, Mozilla Project, Pigdin, Scribus, Sakai, Wikimedia and WordPress were accepted.

**2010** Suggest that we work with appropriate successful 2009 GSOC mentoring organisations on project proposals. Encourage them to mentor T<sub>E</sub>X-related projects.

## What next: new skills

To rejoin the mainstream, we may need to learn some new skills and languages.

- ▶ AJAX
- ▶ JavaScript
- ▶ JSON and JSONP
- ▶ Social websites
- ▶ Unicode
- ▶ Web frameworks (such as Django)
- ▶ XML

We may also need to communicate better: are you interested in a monthly electronic newsletter on technical aspects of mathematical content?

# UK Math Content Workshop - Wed 9th Sept 2009

One-day workshop on technical issues related to mathematical content in electronic media. Three main themes

1. Content related technical problems in supporting eLearning in mathematics
2. Standards related to digitisation of mathematics research literature
3. Formulas and equations in otherwise non-mathematical content

The aims of the workshop are improved practice, more collaboration and reuse of software, and a published roadmap to inform and guide future work in this area.

To take place at The Open University, Milton Keynes, UK. Possible due to funding from JISC and the OU. Organised by Jonathan Fine (OU), David McKain (Edinburgh) and Petr Sojka (Masayrk).