

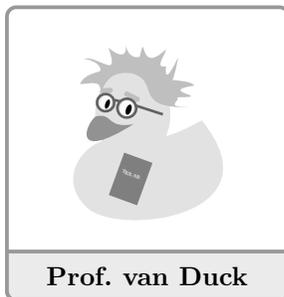
## The DuckBoat — News from T<sub>E</sub>X.SE: Asking effective questions

Herr Professor Paulinho van Duck

### Abstract

Prof. van Duck would like to keep you up to date about the latest topics discussed in the chat of T<sub>E</sub>X StackExchange (T<sub>E</sub>X.SE), the famous Questions & Answers site dedicated to T<sub>E</sub>X, L<sup>A</sup>T<sub>E</sub>X and friends. For this installment, he would also like to show you how to ask a good question on the same site, namely, how to have a rapid and smart answer by asking in the correct way.

### 1 Pleased to meet you!



Hi, T<sub>E</sub>X/L<sup>A</sup>T<sub>E</sub>X friends!  
I am Herr Professor Paulinho van Duck, if you usually attend the T<sub>E</sub>X.SE chat, you likely know me already.

I was born in São Paulo, Brazil, but now I live in Milan, Italy. I share a flat with my friend Carla, known as

CarL<sup>A</sup>T<sub>E</sub>X on T<sub>E</sub>X.SE. She also helps me writing my documents and managing my emails: typing is difficult when you do not have a pointy beak, quack!

I was named *Paulinho* after Paulo Cereda,<sup>1</sup> the author of the very convenient `arara` tool. He is developing a new version of it — we all are looking forward to using it.

Ulrike Fischer added the *van* before my surname as an allusion to the Dutch (not *Duck*) colonization of Brazil. I take this opportunity to thank her and her husband for making me become a donor for Mönchengladbach zoo.

Barbara Beeton — I think she needs no introduction — gives me the *Herr Professor* title, in order to make it (ridiculously) more formal.

Last but not least, I would like to thank samcarter for creating the `tikzducks` package;<sup>2</sup> my image and the other ducks you will find here are drawn by it.



I am a newbie with L<sup>A</sup>T<sub>E</sub>X, but I am very enthusiastic about it. I have read that beginner's level articles on this journal are very welcome. Hence, here I am!

<sup>1</sup> His advice about how to write this article was invaluable.

<sup>2</sup> <https://ctan.org/pkg/tikzducks>.

This is the first duck-column. If you like it, maybe there will be others. Do not hesitate to email me for suggestions about new topics (I have already in mind a TikZ Quack Guide) or for any criticisms.

### 2 Quacking in chat

If you haunt any programmers' milieux, you have probably already heard about the famous *editor war*: the rivalry between Emacs and Vim users always gives life to lively conversations also in our chat!

Recently a new war has come up, more or less with the same savagery: Italians vs. Rest of the World about the so-called *pineapple pizza*.

Since I live in Italy, I must say that that *thing* is not a *pizza* (also because, otherwise, Carla will throw me out). However, there are a lot of people, all around the world, who like these strange matched flavors.

If you join our chat in a quiet moment, just mention this subject to get a lively conversation.



Early this year, while we were quietly making fun of Paulo because of his never-ending thesis, my colleague Enrico Gregorio (`egreg`) shocked us by posting a question on T<sub>E</sub>X.SE Meta. It happened that double backslashes disappeared from the code of some posts. If you know (L<sup>A</sup>)T<sub>E</sub>X at all, you will surely know how important backslashes are.

The issue was due to some software update by StackExchange, dated back to 2013, and it was impossible to address in an automated way.

Hence, David Carlisle, Moriambar, Barbara Beeton, and many others manually (or semi-manually, with some JavaScript) fixed the codes in about 10,000 questions and answers. We are grateful to all of them!

Now that all the errors are fixed, finally David can return to complaining about Enrico's stealing his ticks.



Another hot (even in the meteorological meaning) topic is the next T<sub>E</sub>X Users Group meeting in Rio de Janeiro, Brazil.

Paulo Cereda is preparing some little (or should I say big?) surprises. Knowing him (and looking at the pictures<sup>3</sup> he posted in chat), I think the next conference will be the funniest one ever organized.

There will also be a party at the famous Copacabana Beach and you will be able to enjoy the breath-taking sunsets at Ipanema!

<sup>3</sup> From Twitter: <https://twitter.com/paulocereda/status/876420672683167745>.



Figure 1: T<sub>E</sub>X Users Group Meeting 2018 in Brazil.

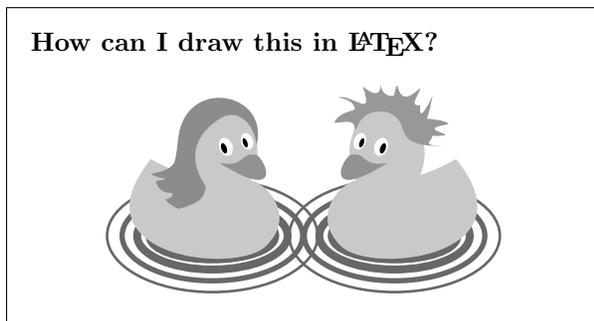
Therefore, prepare your bathing suit (if you have the *physique du rôle*, you can even dare a thong) and buy a flight to Brazil!

### 3 Quack Guide No. 1

#### Asking *effective* questions on T<sub>E</sub>X.SE

##### 3.1 How you should *not* ask

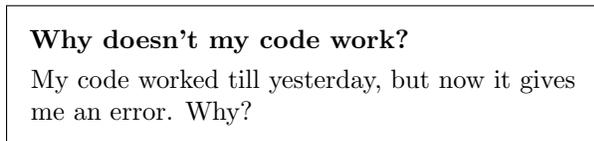
Now and then, a question like the following appears on T<sub>E</sub>X.SE:



Its flaws are (or should be) obvious: a generic title, followed by a more or less complicated picture (or table or whatever) without either an explanation of the problem or a single line of code.

We call these questions *just-do-it-for-me*, because they do not show any effort by the OP (i.e. the Original Poster, in T<sub>E</sub>X.SE jargon). Please, *never* post such stuff, a little duck cries when he sees it!

Another type of *bad* question is something like:



How can we answer that? Do you think we have a crystal ball? Or that we can read your mind?

Herr Professor Paulinho van Duck

What is the error? Where is an example of your code?



On the contrary, a smart user always follows these guidelines:

#### Van Duck's rules

1. Read the package manuals.
2. Look at the log generated by the code.
3. Search on T<sub>E</sub>X.SE and, in general, on the Internet.
4. Add a minimal working example (MWE) to your question.

The last point deserves to be examined in depth. Only a very few questions do not need a minimal working example, which is to say, a complete (but as short as possible) document which reproduces your problem. For example, if you are asking how to set a feature of your editor, probably an MWE is useless. For all the other ones, it is *indispensable*!

Nevertheless, it often happens that it is not added, especially by newbies.

My in-depth study of this phenomenon has led me to state:

#### Van Duck's equation

$$U_b = U_k + U_h + U_l$$

where the users who ask a *bad* question without an MWE ( $U_b$ ) are divided into three categories:

$U_k$  = who *do not know* what an MWE is

$U_h$  = who vaguely know what an MWE is but do not know *how* to create it in a correct way

$U_l$  = who know what an MWE is but are too *lazy* to add it.

Needless to say, we are particularly annoyed by the last type. For the rest of us, I hope this duck-column may be useful.



On T<sub>E</sub>X.SE there are a lot of pages which can help you with the asking process (see Table 1). I will try to sum them up very briefly in the next subsections. I would also like to suggest to you some little tricks, probably unknown by beginners.

##### 3.2 What you should do *before* asking

One of the pluses of T<sub>E</sub>X & Co. is the abundant documentation — take advantage of it.

**Table 1:** Useful T<sub>E</sub>X.SE pages concerning the asking-a-question process

No.	URL	Description
1.	<a href="http://tex.meta.stackexchange.com/q/1436">tex.meta.stackexchange.com/q/1436</a>	Starter guide
2.	<a href="http://tex.stackexchange.com/q/162">tex.stackexchange.com/q/162</a>	List of online resources about T <sub>E</sub> X, L <sup>A</sup> T <sub>E</sub> X and friends
3.	<a href="http://tex.stackexchange.com/help/searching">tex.stackexchange.com/help/searching</a>	How to search within T <sub>E</sub> X.SE
4.	<a href="http://tex.stackexchange.com/help/how-to-ask">tex.stackexchange.com/help/how-to-ask</a>	How to ask a good question
5.	<a href="http://tex.meta.stackexchange.com/q/228">tex.meta.stackexchange.com/q/228</a>	What a minimal working example (MWE) is
6.	<a href="http://tex.meta.stackexchange.com/q/4407">tex.meta.stackexchange.com/q/4407</a>	How to create a minimal working example with bibliography (MWEB)
7.	<a href="http://tex.meta.stackexchange.com/q/1852">tex.meta.stackexchange.com/q/1852</a>	How to accept an answer

I know that reading the package manuals could be boring, sometimes even impossible. (I think nobody on earth has read all of the more than 1,000 pages of the TikZ & PGF manual!)

However, information such as incompatibilities or cautions in loading the package before/after others is usually written at the beginning. A rapid look at the documentation is mandatory!

In part I of the TikZ manual, for example, there is an excellent tutorial. Reading it is enough to start using this monster package.



Another fundamental tool is the log, first of all in the case of errors.

OK, T<sub>E</sub>X error descriptions are not the ultimate in clarity. A beginner is usually astonished when s/he discovers that *Undefined control sequence* merely means that a command is misspelled or the package it belongs to is not loaded.

However, if you search for those mysterious error messages on the Internet, almost always you find the solution or, at least, you understand what they mean.

In the log, there is also the position where the problem was detected, which (almost always) corresponds to the line of the wrong instruction.

Eventually, remember that the most important error is the first one; the others may be a consequence of that one.

But I don't want to teach you how to debug your code, there is already a gorgeous article by Barbara Beeton about it (*TUGboat* 38:2, p. 159, [tug.org/TUGboat/tb38-2/tb119beet.pdf](http://tug.org/TUGboat/tb38-2/tb119beet.pdf)).

Instead, I would like to show you an effective command: `\listfiles`. It writes on your log the versions of all the packages you are using. It is useful in cases like: *Why does this code work on my friend's computer but not on mine?* or *Why does it work on ShareL<sup>A</sup>T<sub>E</sub>X but not on Overleaf?* Simply because there are different package versions. Many problems could be solved merely by updating your T<sub>E</sub>X distribution.

To see how it works, run this simple document:

```
\listfiles
\documentclass{article}
\usepackage{tikzducks}
\begin{document}
  \begin{tikzpicture}
    \duck
  \end{tikzpicture}
\end{document}
```

Then, in your log, within

```
*File List*
...
*****
```

you will find many packages listed, with their version besides. Note that, in my example, I have loaded only `tikzducks` but the log shows all the packages loaded by `tikzducks` itself.

It allows you to avoid loading packages twice, unless you need to load them with a particular option, and it is also useful to detect hidden incompatibility.



Another huge source of information is the Internet. If you have a problem, it is very likely that someone else had the same problem before; search with your browser, an answer will probably appear. This may seem trivial, but I assure you that many times I've seen questions on T<sub>E</sub>X.SE which could have been rapidly solved that way, quack!

However, pay attention: like everything you find on the Internet, some information could be incorrect or obsolete. For instance, `\rm` in math has been deprecated for more than twenty years (you should use `\mathrm` instead) but it appears here and there on the net. Look at a couple of online resources before deciding to use anything. A list of reliable ones can be found in the post indicated in Table 1, No. 2.

You can also directly use the search field on the top right of T<sub>E</sub>X.SE main site home page (see Table 1, No. 3). Again, this may seem trivial, but every day we close questions because they are a duplicate of other ones!

### 3.3 How you *should* ask

If all your searching was fruitless, let us see how to ask a question.

First of all the title: be specific! Do not write things like *How can I do this in L<sup>A</sup>T<sub>E</sub>X?* or *Why doesn't my code work?* Many users read the questions only if they think (looking at the title) that they could answer. If your title is generic or unclear, you may miss the chance of having a prompt answer. Moreover, future users with the same problem as yours will not be able to easily find your post.

Second, in the body of your question give all the details needed to understand your problem. Don't put only the code or, even worse, only an image.

Third, for questions which need it (I dare say about 99.9% of the total on T<sub>E</sub>X.SE), add the most important thing: the MWE. Let us rapidly see the essential steps to follow to create it (for more details see Table 1, Nos. 5 and 6).



Do not forget your `\documentclass`. It is important to indicate it, even if you are asking something about a `\tikzpicture` or a math expression. Many things can change if you are using `beamer` instead of `article`, for example.

Then, the packages: list all the packages needed to reproduce your problem, and only those, do not be verbose!

The same for your code. Put it within

```
\begin{document}
...
\end{document}
```

and add all and only the lines strictly necessary to reproduce your problem. Do not post only code snippets.

Remember to test your MWE before adding it to your question! You have to be sure it works or, if it does not work, it gives the same error you are struggling with.

It is useful not only for the ones who would like to answer but — believe me — also for you. I do not know how many times, while I was building my MWE, I found the solution by myself!

There are also many packages which help you to create an MWE.

For instance, `lipsum`<sup>4</sup> and `blindtext`<sup>5</sup> help you to produce some text with no meaning, just to fill your pages, preserving your privacy or copyright. `graphicx`<sup>6</sup> allows you to use some example images. There is also an `mwe`<sup>7</sup> package — guess what it is for!

Moreover, a testing-purpose one is `showframe`,<sup>8</sup> which shows a simple diagram of the page layout. It is handy for detecting the infamous `Overfull \hbox` and for refining alignment in general. You have just to load it in your preamble and compile.

For example, this code gives an `Overfull \hbox`:

```
\documentclass{book}
\usepackage{showframe}
\usepackage{mwe}
\begin{document}
  \blindtext

  \includegraphics[width=\linewidth]{%
    example-image-a}

  \blindtext
\end{document}
```

If you run it, you will get the output shown in Figure 2. As you can see, the indentation error is clearly visible.



If your problem concerns bibliographies, you should also include your `.bib` file in your MWEB (minimal working example with bibliography). Of course, you do not have to include the complete file, but only the bibitems which cause your trouble.

The best practice is to create an MWEB with your `.bib` file embedded (in this way, people who would like to help you just have to cut and paste your code and compile it, and thus increase the probability of getting a quick answer).

The `filecontents*` environment<sup>9</sup> is your friend in building such an MWEB.

It is also preferable, instead of inventing a new name for your test `.bib` file, to use `\jobname.bib`. This automatically uses the same name as the `.tex` file, changing only the suffix.

Here is a simple scheme you could follow:

<sup>4</sup> <https://ctan.org/pkg/lipsum>.

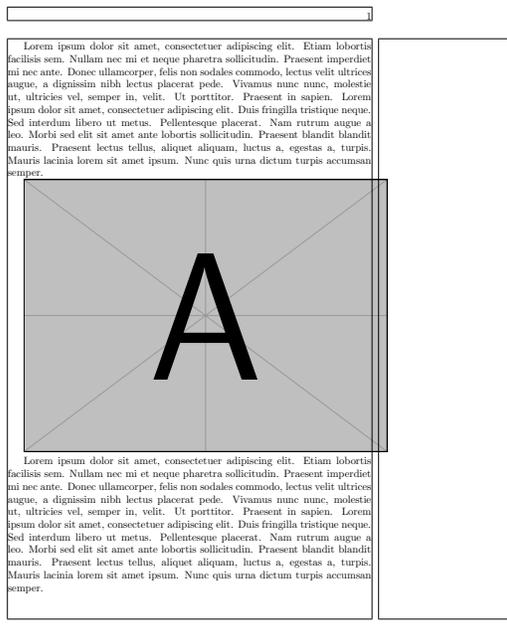
<sup>5</sup> <https://ctan.org/pkg/blindtext>.

<sup>6</sup> <https://ctan.org/pkg/graphicx>.

<sup>7</sup> <https://ctan.org/pkg/mwe>.

<sup>8</sup> <https://ctan.org/pkg/showframe>.

<sup>9</sup> There is also a `filecontents` package, <https://ctan.org/pkg/filecontents>.



**Figure 2:** An example of usage of the `mwe` and `showframe` packages. The `Overfull \hbox` is immediately identifiable.

```

\begin{filecontents*}{\jobname.bib}
% put your bibitems here
...
\end{filecontents*}
\documentclass{...}
% put your packages here
% including the bibliographic one
...
% with biblatex:
\addbibresource{\jobname}
\begin{document}
% put here your code with your citation
...
% and bibliography printing, with biblatex:
\printbibliography
% or with other bibliographic packages:
\bibliography{\jobname}
\end{document}

```

Alternatively, if your problem is not strictly concerned with your own bibitems, you could use a test file included in the powerful `biblatex`<sup>10</sup> package: `biblatex-examples.bib`.



<sup>10</sup> <https://ctan.org/pkg/biblatex>.

Eventually, to complete your question, you have to add the correct tags. Again, there are users who read a question only if it is tagged in a way they think they could answer. Hence, tags are essential.

Look at the tag description before using it. For example, `latex3` concerns new material being developed by the `LATEX3` project, but it is often *wrongly* used as a generic tag for `LATEX` questions.

### 3.4 What you should do *after* asking

Once you have posted your question, your work is not over.

You have to pay attention to the possible comments of other users and reply to them; usually they ask for clarifications.

Finally, when someone posts the answer with the solution you were waiting for, you have to take a very important action: accept it by clicking the specific tick (see Table 1, No. 7). Remember that the users who answered are not paid for it; accepting the best answer is the correct way to say *Thank you!*

If you have more than 15 reputation points (and it is straightforward to get them if you post good questions), you can also upvote all the answers which are useful for you. Please do it, quack!

**How to thank**



1

Click here to *upvote* any useful post



Click here to *accept* the best answer to your question

## 4 Conclusions

I hope you liked my explanation, and if you have trouble in asking a question, remember:

*You are lucky, there's a ducky!*

- ◇ Herr Professor Paulinho van Duck  
Quack University  
Sempione Park Pond, Milano  
Italy  
paulinho dot vanduck (at) gmail dot com