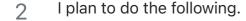


PDF comparison to test one package

Asked today Modified today Viewed 41 times



Do any of you use PDFs to test the good outputs of their packages?





(1)

- 1. Build by hand examples that I indicate to be correct: TEX + PDF files.
- 2. Each time I update my source code, I recompile the TEX files of the first step to compare their PDFs to those marked as correct.

Do you do this? Is there an existing script that already does this?

Share Edit Close Delete Flag

edited 8 hours ago

projetmbc

asked 8 hours ago

3 ▲ 13build can do this, although typically testing using the .log is better. I think we need more detail to provide a solid answer – Joseph Wright ♦ 8 hours ago

I need to check that no regression has been done in the output of, for example, tables of variation, or probabilistic trees... This is for visual outputs. – projetmbc 7 hours ago

Delete

it is much more reliable to diff the log, add \showoutput so the log has a symbolic representation of the output. I3build will automate normalising away paths and times. This is how the latex test suite has worked for over 30 years – David Carlisle 6 hours ago

@DavidCarlisle Is there some demos, or tutorials showing how to think like a (La)TeX tester? - projetmbc 6 hours ago

- 2 texdoc l3build or look at the test files in any of the latex3 or ho-tex github repositories David Carlisle 6 hours ago
- 1 <u>tug.org/TUGboat/tb36-3/tb114wright.pdf</u> or look at <u>github.com/josephwright/siunitx</u> for an example using 13build for a smaller stand-alone package <u>Joseph Wright</u> ♦ 3 hours

\$

1 Answer

Sorted by: Highest score (default)



This can be automated with I3build



with a directory layout containing



```
build.lua
pkg.sty
testfiles/test1.lvt
```



build.lua

```
module = "pkg"
sourcefiles={"*.sty"}
```

pkg.sty

\ProvidesPackage{pkg}[2022-11-19 l3build example package]

\def\foo{abc}

testfiles/test1.lvt

\end{document}

```
\documentclass{article}
\usepackage{pkg}
\input{regression-test}

\begin{document}
\START

\TEST{basic expansion}{\show\foo}
\showoutput

abc \foo
\foo
```

You can automate running the test.

build.lua sets up the package, I need to specify the .sty is source here as normally it assumes .sty are generated from .dtx sources.

The test file is a latex document, but conventionally given .1vt extension. You can test in various ways, here I use \show but also \showoutput which shows a symbolic representation of the pdf output.

Then

```
13build save test1
```

will run latex and save a normalised test result log file as test1.tlg Assuming this looks good this would normally be checked in to source control.

13build check

will then run all tests

\$ 13build check

```
Running checks on
test1 (1/1)
--> failed

Check failed with difference files
- ./build/test/test1.luatex.diff
- ./build/test/test1.xetex.diff
```

as luatex uses latin modern opentype and our saved result showoutput is using computer modern, so save engine specific reference results:

```
cp ./build/test/test1.luatex.log testfiles/test1.luatex.tlg
cp ./build/test/test1.xetex.log testfiles/test1.xetex.tlg
```

Then check again:

```
$ 13build check
Running checks on
  test1 (1/1)
All checks passed
```

Now if we make a change to the package

```
\ProvidesPackage{pkg}[2022-11-19 13build example package]
\def\foo{
% egreg left a space
abc}
```

and run check:

```
$ 13build check
Running checks on
test1 (1/1)
--> failed

Check failed with difference files
- ./build/test/test1.luatex.diff
- ./build/test/test1.pdftex.diff
- ./build/test/test1.xetex.diff
```

Then you need to check the diffs and revert the change, or update the tlg files.

Share Edit Follow Flag



Great. I like that kind of automation. Thanks for this big small example! Hoping that egreg will not see your example. :-) - projetmbc 5 hours ago