`tikzinput`: Selective Input of TIKZ Pictures*

Michael Kohlhase
Jacobs University, Bremen
`http://kwarc.info/kohlhase`

November 12, 2015

Abstract

Running `tikz` takes a lot of time in LaTeXML, therefore it is often more efficient externalize the TIKZ pictures into separate (standalone) files, to let LaTeX handle the TIKZ pictures to generate an image, and just load it via the usual LaTeX graphics packages. The `tikzinput` package supports this workflow, and allows to switch back to native TIKZ via a package option.

Contents

*Version v1.0 (last revised 2015/10/22)

1 Introduction

Running `tikz` takes a lot of time in LaTeXML, therefore it is often more efficient externalize the TIKZ pictures into separate (standalone) files, to let LaTeX handle the TIKZ pictures to generate an image, and just load it via the usual LaTeX graphics packages. The `tikzinput` package supports this workflow, and allows to switch back to native TIKZ via a package option.

A side-effect of the workflow described above is that the TIKZ pictures can be developed – and formatted – independently of the document they are intended for. They can essentially be treated like an image file, which can be included into multiple documents.

2 The User Interface

2.1 Package Options

`image` The behavior of the `tikzinput` package is determined by whether the `image` option is given. If it is not, then the `tikz` package is loaded, all other options are passed on to it and `\tikzinput{`⟨*file*⟩`}` inputs the TIKZ file ⟨*file*⟩`.tex`; if not, only the `graphicx` package is loaded and `\tikzinput{`⟨*file*⟩`}` loads an image file ⟨*file*⟩`.`⟨*ext*⟩ generated from ⟨*file*⟩`.tex`.

2.2 Inputting Standalone TIKZ Pictures

The selective input functionality of the `tikzinput` package assumes that the TIKZ pictures are externalized into a standalone picture file, such as the one Example 1.

```
\documentclass{standalone}
\usepackage{tikz}
\usetikzpackage{...}
\begin{document}
  \begin{tikzpicture}
     ...
  \end{tikzpicture}
\end{document}
```

Example 1: A Standalone TIKZ Picture File

The `standalone` class is a minimal LaTeX class that when loaded in a document that uses the `standalone` package: the preamble and the `document` environment are disregarded during loading, so they do not pose any problems. In effect, an `\input` of the file in Figure 1 only sees the `tikzpicture` environment, but the file itself is standalone in the sense that we can run LaTeX over it separately, e.g. for generating an image file from it.

This is exactly where the `tikzinput` package comes in: it supplies the `\tikzinput` `\tikzinput` macro, which – depending on the `image` option – either directly inputs

the TIKZ picture (source) or tries to load an image file generated from it.

Concretely, if the `image` option is not set for the `tikzinput` package, then `\tikzinput[`⟨*opt*⟩`]{`⟨*file*⟩`}` disregards the optional argument ⟨*opt*⟩ and inputs ⟨*file*⟩`.tex` via `\input`. If it is, `\tikzinput[`⟨*opt*⟩`]{`⟨*file*⟩`}` expands to `\includegraphics[`⟨*opt*⟩`]{`⟨*file*⟩`}`.

3 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the sTeX GitHub repository [sTeX].

1. none reported yet

4 Implementation

General Setup for LaTeXML:

```
⟨*ltxml⟩
# -*- CPERL -*-
package LaTeXML::Package::Pool;
use strict;
use LaTeXML::Package;
use Unicode::Normalize;
⟨/ltxml⟩
```

4.1 Package Options and Required Packages

\iftikzinput@image

We define a new switch `\iftikzinput@image` and the `image` option. Apart from that we accept all options that might come our way.[1]

EdN:1

```
⟨*package⟩
\newif\iftikzinput@image\tikzinput@imagefalse
\DeclareOption{image}{\tikzinput@imagetrue}
\DeclareOption*{}
\ProcessOptions
⟨/package⟩
⟨*ltxml⟩
DeclareOption('image', sub { AssignValue('tikzinput@image' => 1); });
DeclareOption(undef,sub {PassOptions('tikz','sty',ToString(Digest(T_CS('\CurrentOption'))));});
ProcessOptions();
⟨/ltxml⟩
```

Next we require the packages we need, in the `image` case, we have to also provide "empty" versions of some TIKZ macros and environments that do not get defined as the `tikz` package is not loaded.

```
⟨*package⟩
\iftikzinput@image
\RequirePackage{graphicx}
\providecommand\usetikzlibrary[1]{}
\else
\RequirePackage{tikz}
\RequirePackage{standalone}
\fi
⟨/package⟩
⟨*ltxml⟩
if (LookupValue('tikzinput@image')) {
RequirePackage('graphicx');
RawTeX(' \providecommand\usetikzlibrary[1]{}'); }
else {
RequirePackage('tikz');
RequirePackage('standalone');}
⟨/ltxml⟩
```

[1] EdNote: MK: Actually we would have liked to pass all options to TIKZ, but that does not work, since that is specific about its options.

4.2 Inputting Standalone TIKZ Pictures

`\tikzinput` Depending on the `image` option, we do the necessary things.

```
⟨*package⟩
\iftikzinput@image
\newcommand\tikzinput[2][]{\includegraphics[#1]{#2}}
\else
\newcommand\tikzinput[2][]{\input{#2}}
\fi
⟨/package⟩
⟨*ltxml⟩
if (LookupValue('tikzinput@image')) {
DefMacro('\tikzinput[] {}','\includegraphics[#1]{#2}');}
else {DefMacro('\tikzinput[] {}','\input{#2}');}
⟨/ltxml⟩
```

`\*tikzinput` The variants we define in terms of `\tikzinput`.

```
⟨ltxml⟩RawTeX('
⟨*package | ltxml⟩
\newcommand\ctikzinput[2][]{\begin{center}\tikzinput{#2}\end{center}}
⟨/package | ltxml⟩
⟨ltxml⟩');
```

5 Finale

We need to terminate the file with a success mark for perl.

```
⟨ltxml⟩1;
```

References

[sTeX] *KWARC/sTeX*. URL: `https://github.com/KWARC/sTeX` (visited on 05/15/2015).