Advances in Python\TeX with an

Geoffrey Poore

Union University
Jackson, TN
Advances in Python\LaTeX\ with an introduction to \texttt{fvextra}

Geoffrey Poore
Union University
Jackson, TN
Background
Python\TeX\ (2011)

\usepackage{pythontex}
...
\begin{pycode}
print("Hello from Python!")
\end{pycode}
\usepackage{pythontex}
...
\begin{pycode}
print("Hello from Python!")
\end{pycode}

pdflatex document.tex
pythontex document.tex
pdflatex document.tex
\usepackage{pythontex}
...
\begin{pycode}
print("Hello from Python!")
\end{pycode}

pdflatex document.tex
pythontex document.tex
pdflatex document.tex

Hello from Python!
Background
Python\TeX (2011)

\begin{pyblock}
x = 2**8
print("Hello from Python!")
\end{pyblock}

\printpython\textex The variable \(x = \text{\texttt{x}}\).
The variable $x = \texttt{x}$.

\begin{pyblock}
x = 2**8
print("Hello from Python!")
\end{pyblock}

\printpython\texttt{x} The variable $x = \texttt{x}$.

\begin{pyblock}
x = 2**8
\texttt{print}("Hello from Python!")
\end{pyblock}

Hello from Python! The variable $x = 256$. 

$\begin{align*}
\texttt{x} &= 2**8 \\
\texttt{print}("Hello from Python!")
\end{align*}$
\begin{pycode}
from pylab import *
figure(figsize=(4,3))
x = linspace(0, 4, 1001)
plot(x, 2*sin(2*pi*x/4))
xlabel('x (m)')
ylabel('y (m)')
grid(True)
savefig('wave.pdf', bbox_inches='tight')
\end{pycode}

\includegraphics[scale=0.4]{wave}
Background
Python\TeX{} (2011)

Also options to typeset code with no execution (Pygments syntax highlighting)

- \texttt{pyverbatim} environment and \texttt{\pyv} command
- \texttt{pygments} environment and \texttt{\pygment} command that follow minted syntax
Background
PythonTEX (2011)

Also options to typeset code with no execution (Pygments syntax highlighting)

- \texttt{pyverbatim} environment and \texttt{\pyv} command
- \texttt{pygments} environment and \texttt{\pygment} command that follow minted syntax

Support for executing other languages
- Ruby, Julia, Octave, Sage, Bash, Rust, ...
Verbatim for code

fancyvrb

Timothy Van Zandt, Herbert Voß, Denis Girou, Sebastian Rahtz, Niall Mansfield

v2.0 Beta 1994/03/30 First version personally shown by Timothy van Zandt
v2.5 1998/01/28 First public release.
v2.6 1998/07/17 Three bug corrected, options numberblanklines, label, labelposition and leftline added and few precisions.
v2.7 2000/03/21 DG/SR changed how fancyvrb.cfg included
v2.7a 2008/02/07 NMM fixed lastline=firstline bug
v2.8 2010/05/15 fixed bug with several trailing spaces (hv)

Used in minted, pythontex, listings (optionally), ...
Verbatim for code

fvextra

v1.0 2016/06/28 initial release

- Extends and patches fancyvrb
- New features fully supported in pythontex and also minted
- Most features will work in other packages with no modification
fvextra
Quotation marks — upquote by default

\begin{Verbatim}
`Single quoted text' ``Double quoted text''
\end{Verbatim}

`Single quoted text' ``Double quoted text''

\begin{Verbatim}[curlyquotes]
`Single quoted text' ``Double quoted text''
\end{Verbatim}

‘Single quoted text’ "Double quoted text"
fvextra
Math in verbatim — closer to normal math mode

\begin{Verbatim}[commandchars=\\\{\}, mathescape]
\$x^2 + \frac{d}{dx} f_{\text{sub}}(x) = g(x)\$
\end{Verbatim}

fancyvrb: \(x^2 + \frac{d}{dx} f_{\text{sub}}(x) = g(x)\)

fvextra: \(x^2 + \frac{d}{dx} f_{\text{sub}}(x) = g(x)\)
\fvset{obeytabs, showtabs,  
tab=\rightarrowfill, tabcolor=blue} 
\begin{pyverbatim} 
some_string = '' 
  # First line 
  # Second line 
  #'' 
\end{pyverbatim}
\ifnum\catcode`\{=1
\errmsg
{LaTeX must be made using an initex with no format preloaded}
\fi
\catcode`\{"=1
\catcode`\}==2
\ifx\directlua\undefined
\else
  \ifx\luatexversion\undefined
    \directlua{tex.enableprimitives("",%tex.extraprimitives('etex', 'pdftex', 'umath'))}
    \directlua{tex.enableprimitives("",%tex.extraprimitives("omega", "aleph", "luatex"))}
  \fi
  \fi
\fi
fvextra
breaklines

[breaklines=false] (default)
fvextra
breaklines

[breaklines=true]

\ifx\directlua\undefined
  \else
    \ifx\luatexversion\undefined
      \directlua{tex.enableprimitives('',%}
      tex.extraprimitives('etex',
                      \leftarrow 'pdftex', 'umath'))}
      \directlua{tex.enableprimitives('',%}
      tex.extraprimitives('omega',
                      \leftarrow 'aleph', 'luatex'))}
    \fi
  \fi
\fi
fvextra
breaklines

\begin{Verbatim}[breaklines,  
  breaksymbolleft=\ding{43},  
  breaksymbolright=\breaksym]  
  A very long line of text that just went right off the slide  
\end{Verbatim}
fvextra
breakbefore and breakafter

\begin{Verbatim}[breaklines, breakbefore=C, breakafter=T]
TATACCATGTGATTCATTTTACTTGATTTAACAATAAAAAATATAAATACATTGTAATTCATTTTTGGTAACCATTTCA
\end{Verbatim}

TATACCATGTGATTCATTTTACTTGATTTAACAATAAAAAAT

\rightarrow
ATAAATAACATTGTAATTCATTTTTGGTAACCATTTCA

\rightarrow
CAAAAAGTGTGGGGAAATTAATTTGGGAATTACTCTCCTC

\rightarrow
CATTGAAAAATATCTCATTTTGCTAAAATAAGACAGT

\rightarrow
AAAACAGTACAGTTTAAATATTTATAAAAAT

\rightarrow
AGGAAAGTTTGCGAAAAAGAGAGGAGTACACACCTGTGA

breaks are catcode-independent and by default group identical characters!
Python\TeX

\texttt{\textsubscript{sub} environment, or how to avoid catcodes and other trickery}

\begin{tikzpicture}
\draw (0, 0) -- (2, 0);
\draw (0, 0) -- (1, 1);
\end{tikzpicture}
Python\TeX

sub environment, or how to avoid catcodes and other trickery

\begin{pycode}
from math import *

p = sqrt(2)
\end{pycode}

\begin{tikzpicture}
\draw (0, 0) -- (2, 0);
\draw (0, 0) -- (\py{p}, \py{p});
\end{tikzpicture}
! Missing \endcsname inserted.
<to be read again>
    \xdef
1.472 \draw (0, 0) -- (\py{p}, \py{p})
    ;

The control sequence marked <to be read again> should not appear between \csname and \endcsname.
PythonTeX

sub environment, or how to avoid catcodes and other trickery

\begin{pycode}
from math import *
p = sqrt(2)
template = '''
\\begin{{{tikzpicture}}}
\\draw (0, 0) -- (2, 0);
\\draw (0, 0) -- ({p}, {p});
\\end{{{tikzpicture}}}
'''

print(template.format(p=p))
\end{pycode}
Python\TeX

sub environment, or how to avoid catcodes and other trickery
Python \TeX

\texttt{sub} environment, or how to avoid catcodes and other trickery

\begin{pycode}
from math import *
p = sqrt(2)
\end{pycode}

\begin{pysub}
\begin{tikzpicture}
\draw (0, 0) -- (2, 0);
\draw (0, 0) -- (!{p}, !{p});
\end{tikzpicture}
\end{pysub}
Python\TeX

\sub environment, or how to avoid catcodes and other trickery
Python\TeX

sub command, or how to avoid catcodes and other trickery

\begin{pycode}
x = 2**16
\end{pycode}

\pys{\verb|x = \{x\}|}

x = 65536
Currently, edit Python code and create classes

bash_template = ''
    cd "{workingdir}"
    {body}
    echo "{dependencies_delim}"
    echo "{created_delim}"
''

bash_wrapper = ''
    echo "{stdoutdelim}"
    >&2 echo "{stderrdelim}"
    {code}
''

bash_sub = '''echo "{field_delim}"
    echo {field}"
'''

CodeEngine('bash', 'bash', '.sh',
    '{bash} "{file}.sh"',
    bash_template, bash_wrapper, '{code}', bash_sub,
    ['error', 'Error'], ['warning', 'Warning'],
    'line {number}')

Soon, just drop a config file in pythontex/languages
Limited support for interactive programs coming “soon”

Requirements

- Communicate via pipes
- Predictable encoding
Don’t typeset code without \fvextra!