Things \TeX{} should (probably) not do

Stephen Hicks

Cornell University

Bacho\TeX{} 2009, 3 May 2009
\textbf{inline_def: motivation}

\texttt{\expandafter \expandafter \expandafter \newtheorem \expandafter \expandafter \expandafter \expandafter \expandafter \expandafter \expandafter \begin{sthshadeenvname} \expandafter \expandafter \expandafter \end{sthshadeenvname} \begin{sth@caption} \[ \begin{sth@within} \]}

\texttt{may be better expressed as}

\texttt{\ExpandSome{\newtheorem{\Expand\sth@shadeenvname}\Expand\sth@caption}[\Expand\sth@within]}
inlinedef: further motivation

extending definitions

\expandafter\def\expandafter\foo\expandafter{\foo\bar}

compare with

\texttt{\texttt{Inline}\def\foo{\Expand\foo\bar}}

• this is not trivial: consider expanding across, e.g. #1.
• often better than the common \texttt{let\old@foo\foo} approach:
  • does not pollute namespace
  • eliminates problems from conflicting names
more generally...

recursive token scanning

- how can we scan arbitrary tokens losslessly?
- difficulties:
  - groups can be explicit or implicit
  - spaces can get lost
  - how to implement callbacks (many options...)

\def\scan\{\futurelet\foo\switch
\def\switch{%
  \let\next\normal
  \ifcat\noexpand\foo\space \let\next\dospace\fi
  \ifcat\noexpand\foo\bgroup \let\next\trygroup\fi
  \ifcat\noexpand\foo\relax \trycb{&\meaning\foo}\fi
  \next}
\let\expandafter
\long\def\normal#1{\toks0\expandafter\the\toks0 \#1}\scan}

\def\dospace{\toks0\expandafter\the\toks0 \space}\ea\def\ea\unspace\space{}{}

\long\def\trygroup#1#{% 
\def\temp{#1}\ifx\temp\empty\ea\recurse\else\ea\normal\fi#1%
\long\def\recurse#1{% 
\begingroup\toks0{}\scan#1\END{}\ea\endgroup\ea 
\toks\ea0\ea\ea\ea{\ea\the\ea\toks\ea0\ea{\ea\the\ea\toks0}}\scan}

\def\trycb#1{\ifcsname #1\endcsname \let\ea\next\csname #1\endcsname\fi} 
\def\callback#1#2#{\def#1{\noexpand#1}%( \ea\def\csname&\meaning#1\endcsname#2} \callback\END#1{}}
usage

\callback\EXPAND#1{\expandafter\scan}
\def\baz{!}
\scan foo {bar \EXPAND\baz} \baz \END
\message{\the\toks0}

• output: foo\space {bar\space !}\space \baz
• can define other callbacks, too
• disadvantages:
  • $\sim 20\times$ slower
  • no longer expandible
inlinedef package

- uses a similar token scanner internally
- provides macro \Inline to prepend any \(g)\def command
- provides callbacks, including
  - \UnsafeExpand inserts an \expandafter
  - \Expand isolates and expands the next single token
  - \NoExpand inserts the given token(s) directly
- options for how to deal with macro name inside own definition

Interesting usage: scope control

\begingroup\count2=...\count1=...\count0=...
\Inline\def\locals{\count0\Expand\the\count0}
\expandafter\endgroup\locals
a \TeX build system (with David Roundy)

the problem

- writing lecture notes for a course in Python
- want to keep a single copy of demo code
- want to show output in notes
- want to only re-run Python code when necessary
- why not do it all with \TeX (using \texttt{\write18})?
verbatim-like environment

\begingroup
\lccode`\~`\lowercase{\gdef~{\relax}}
\lccode`\~`\lowercase{\gdef~{\relax}}
\catcode`[1\catcode`\]2
\catcode`{12\catcode`\}12
\catcode`/0\catcode`\012
/long/gdef/xpython#1\end{python}[
 /print[#1/relax]/run[#1]/end[python]]
\endgroup

\def\python{%
 \catcode`\'13\catcode`\'13\catcode`\>:13
 \def\do##1{\catcode`##1=12}\dospecials
 \def\par{\pythonpar}\obeylines\obeyspaces\xpython}
printing python

\newif\ifcont
\let\texpar\par
{\lccode`\~`:\lowercase{\gdef~{:\conttrue}}}\def\lpar{\texpar\ifx\foo\relax\else
\hskip\parindent\ifcont... \else>>> \fi\fi
\ifx\foo\par\contfalse\fi
}

\long\def\print#1{{\contfalse\frenchspacing\tt
\def\pythonpar{\futurelet\foo\lpar}\#1}}
running python

\newcount\outnum \newlinecharʻ\^^J
\long\def\run#1{
  \advance\outnum1 \edef\outfile{out-\the\outnum.png}
  \def\pythonpar{^^J}
  \im@write{temp.py}{\prelude\par#1\par\postlude}
  \immediate\write18{md5sum temp.py > temp.md5}
  \im@read{temp.md5}\hash
  \aux@edef{hash:\outfile}{\hash}
  \expandafter\ifx\csname hash:\outfile\endcsname\hash
    \else\immediate\write18{python temp.py \outfile}\fi
  \includegraphics{\outfile}
```python
>>> from visual import sphere
>>> for i in range(-2,3):
...    pos = (2*i,0,0)
...    c = 0.5-i/4
...    sphere(pos=pos,color=(c,0,1-c))
```
mars rover

- design a real-time controller for a mars rover
- communicates over network socket
- control steering/acceleration
- avoid boulders, craters, and martians
- interpret “sensor” data
- goal: minimize time to reach home base

network protocol

- initialize message:
  \[dx \ dy \ time-limit \ min-sensor \ max-sensor \ max-speed \ max-turn \ max-hd-turn \ ;\]
- telemetry message:
  \[time-stamp \ vehicle-ctl \ vehicle-x \ vehicle-y \ vehicle-dir \ vehicle-speed \ objects \ ;\]
**Mars Rover Control**

- Finite state machine
- 3 acceleration states
- 5 turning states
- a to accelerate, b to brake
- l to turn left, r to turn right

```plaintext
T 0  -- -89.322 4.308 -126.8 0.000 c -79.336 -12.393 4.284 ;
bl;1;
T 100 bl -89.322 4.308 -126.7 0.000 c -79.336 -12.393 4.284 ;
T 200 bl -89.322 4.308 -125.4 0.000 c -79.336 -12.393 4.284 ;
T 300 bl -89.322 4.308 -122.9 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
T 400 bl -89.322 4.308 -119.3 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
T 500 bl -89.322 4.308 -114.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
T 600 bl -89.322 4.308 -109.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
T 700 bl -89.322 4.308 -104.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
T 800 bl -89.322 4.308 -99.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
a;
T 900 -L -89.322 4.308 -94.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 ;
T 1000 -L -89.322 4.308 -89.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 b -44.028 -10.122 3.876 ;
a;
T 1100 -L -89.322 4.308 -84.5 0.000 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 b -44.028 -10.122 3.876 ;
a;
T 1300 aL -89.314 4.269 -74.5 0.400 c -55.510 -27.463 3.955 c -79.336 -12.393 4.284 b -44.028 -10.122 3.876 ;
r;r;r;r;r;
```
... in $\TeX$ (why not?)

- 72 hours later...
- 1181 lines of $\TeX$ code in 11 files
- 100 lines of Perl

(demo)
networking

first priority

- no /dev/tcp on LiveCD → 100 lines of Perl instead
- didn’t know \pdfeoffsettime → extra timing messages (t...)

![](image)

- perl starts \TeX process with stdin/stdout piped
- \TeX prints WANT or HERE:... to send/receive (\read16)
architecture

main loop

\gdef\want{\message{^^AWANT^^A}\wait}
\gdef\wait{\begingroup\activecodes\read16 to\command
  \expandafter\endgroup\command}
\gdef\send#1{\message{^^AHERE: #1;^^A}}
\gdef\mainloop{\want\mainloop}

reaction functions

{\catcode\'T\active \global\letT\telem}
{\let\l\let \catcode\'t\active \global\lt\pulse}
\def\sdef#1{\expandafter\def\expandafter#1\space}
\sdef\telem #1 #2 #3 #4 #5 #6 #7 ;{...}
\sdef\pulse #1 ;{...\logic}
\begin{tikzpicture}[node distance=2cm,>=latex]
  \node (start) {Start};
  \node [below of=start] (run) {Running Mode};
  \node [below of=run, yshift=-1cm] (obstacles) {Obstacles Ahead?};
  \node [below of=obstacles, yshift=-1cm] (turn) {Turn to within 15° of target; $v \rightarrow v_{\text{max}}\cos q$};
  \node [below of=turn, yshift=-1cm] (endr) {};\hspace{4cm}
  \node [right of=run, xshift=2cm] (swerve) {Swerving Mode};
  \node [below of=swerve, yshift=-1cm] (obstacles2) {Obstacles Ahead?};
  \node [below of=obstacles2, yshift=-1cm] (clear) {Original Obstacle Cleared?};
  \node [below of=clear, yshift=-1cm] (endw) {\textbf{Logic}};

  \path [->]
  (start) edge (run)
  (run) edge (obstacles)
  (obstacles) edge node [left] {no} (turn)
  (turn) edge (endr)
  (run) edge [loop below] node [left, align=center] {Original Obstacle Cleared?} (run)
  (swerve) edge (obstacles2)
  (obstacles2) edge node [left] {no} (clear)
  (clear) edge node [left] {yes} (endw)
  (swerve) edge [loop below] node [left] {Original Obstacle Cleared?} (swerve);
\end{tikzpicture}

\begin{verbatim}
\def\turn@Rl{lll}... \def\turnstate{R} \def\turnwant{l}
\end{verbatim}
mathematics

• \texttt{dimen} registers for decimal numbers
  → overflow at $\sim 8000$

• \texttt{fpdivide} routines from Donald Arseneau (1993)

• \texttt{fpsqrt} uses Newton’s method

• \texttt{distance} needs care to prevent overflows!

\[
\sqrt{a^2 + b^2} = a\sqrt{1 + (b/a)^2} = b\sqrt{1 + (a/b)^2}
\]

• approximate trig functions over $[0, \pi/4]$:
  \[
  \sin x \sim x - .12x^2, \quad \cos x \sim 1 - .48x^2, \quad \arctan x \sim x - .2x^2
  \]

other issues

• store obstacles in token registers: linear access

• registers never cleared in submitted version → lag!
visualization

- picture environment when run with \LaTeX
- records all seen objects, traces of rover and martians
- (demo)

debugging

- what’s going wrong?
\multiply\obs@r\obsfactor
ICFP website:
“A number of compilers for common languages will be included”
...sadly, this did not include \TeX.

- assembled near-minimal \TeX distribution: 530 files/dirs, 8MB
- install script sets $LD\_LIBRARY\_PATH$ (for kpse), $TEXMFMAIN$, $PATH$ and runs texhash