The `pcatcode` package

Michael J. Downes
American Mathematical Society

Version 1.04b, 2007/08/17

1 Introduction

The `pcatcode` package provides mechanisms for robustly changing the catcode environment for the duration of a package and restoring it afterwards to whatever it was before the package was loaded.

Motivation: The `amsmath` package (for example), makes a number of assignments using characters such as double-quote and left-quote, because these are hard-wired into the TEX syntax for numbers and other things. If a package that makes double-quote an active character is loaded before `amsmath`, all the statements that use double-quote turn into error messages.

But now! Groan and grumble no more, ye package writers, for putting a magical invocation of the pcatcode package at the beginning of your package file, as shown here, will ensure that your package contents will be processed with completely normal catcodes.

```
\NeedsTeXFormat{LaTeX2e}[1995/12/01]
\@ifundefined{PushCatcodes}{%
  \RequirePackage{pcatcode}\relax\PushCatcodes\NormalCatcodes
}{}
\ProvidesPackage{foo}[2002/04/16 v1.04]
```

It must be followed by a matching `\PopCatcodes` command, typically used in conjunction with `\endinput`.

WARNING: This functionality would work best if it were built into the LATEX2e kernel, but it cannot be usefully added to the kernel now without adversely affecting document compatibility across different systems. This package therefore modifies one or two of the low-level package-loading functions defined by the kernel. This means that there may be trouble if any other package, or some future version of LATEX, changes the definitions of those functions.

NOTE: Packages that do not call `\PushCatcodes \NormalCatcodes` will remain transparent to the external catcode environment in the usual way (i.e., for all characters except the `@` character).

2 Implementation

2.1 Preliminary Remarks

At the document level, we assume that the following characters always have their standard catcode:

`\{}`	for commands and arguments
`%`	for comments
`#`	for use with `\newcommand`
`*`	cannot be changed without breaking star-form commands
`[]`	cannot be changed without breaking optional arguments
`~`	already active, no reason to change catcode

One might think that we could assume hyphen, period (and comma, for European decimal notation) are always catcode 12. Experience has shown, however, that for just about every character there is some plausible application that would be facilitated by making that character active and giving it a complex definition. Surely hyphen and period must stay catcode 12 in order to work in statements like the following?

```
\addtocounter{section}{-1}
\addtolength{\columnwidth}{-1.5cm}
```

Yet hyphen and period are important constituents of ordinary text and therefore special applications arise. For example, what if you would like mdashes to be surrounded by extra space but your authors prefer to continue writing `---` in the accustomed manner?

Theoretically speaking, the `pcatcode` package itself has to guard against the kind of catcode problems that it is intended to circumvent. If you would like a nice little TEXnician's exercise, try your hand, before looking at the code below, at the task that I set for myself: Find the minimal set of catcode assumptions that one has to make before attempting to establish normalcy, where normalcy is defined as the state at the end of `latex.ltx`, just before the last `\makeatother`. This is the state that may normally be expected at the beginning of a documentclass file, if the LATEX format file does not have any extensions (e.g., babel) compiled in.

Ready to look, now?

In order for the following code to work it is only necessary that backslash, letters, and digits have their normal catcodes: 0, 11, 12 respectively. One could argue that if `pcatcode.sty` is invoked through a standard `\RequirePackage` call, it can be assumed that curly braces also have their normal catcodes. But I suppose it is not utterly inconceivable that in some special circumstances one might want to load the package with the `\@@input` primitive, sans braces. And in any case the extra overhead to handle braces as well is trivial so for esthetic reasons I'm gonna put it in.

If you look really hard you'll find one or two other assumptions, such as "endlinechar is not a letter". Wait, come to think of it, I can fix that ...

The purpose of the group is chiefly to localize the temporarily changed definitions of `\e`, `\n`, etc.

```
\begingroup\let\e\endlinechar\iffalse\
\fi\chardef\E\e\e13\chardef\n\catcode\e\catcode\e5\relax\relax
\chardef\s\catcode32\chardef\t10\catcode32\t
\chardef\c\catcode37 \catcode37 14 % percent
\chardef\=\catcode61 \catcode61 12 % equal sign
\chardef\l=\catcode123 \catcode123=1 % left brace
\chardef\r=\catcode125 \catcode125=2 % right brace
\chardef\[=\catcode91 \catcode91=12 % left bracket
\chardef\]=\catcode93 \catcode93=12 % right bracket
\chardef\^=\catcode94 \catcode94=7 % hat
\chardef\.=\catcode46 \catcode46=12 % period
\chardef\/=\catcode47 \catcode47=12 % slash
\edef\c{%
  \endgroup
  \def\noexpand\pcat@restore{%
    \catcode\number\e=\number\n \catcode32=\number\s
    \catcode123=\number\l \catcode125=\number\r
    \catcode37=\number\c \catcode61=\number\=%
    \catcode91=\number\[\catcode93=\number\]\catcode94=\number\^%
    \catcode46=\number\.\catcode47=\number\/%
    \endlinechar=\number\E \relax
  }%
}
\c
\endlinechar13\catcode13\string=5\relax\relax
\catcode32\string=10 \catcode37 14\relax\relax
\catcode61 12\catcode123=1\catcode125=2\catcode91=12\catcode93=12\relax
\catcode46=12\catcode47=12\catcode94=7\relax
```

This code was postponed until now to avoid all but the most essential assumptions.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{pcatcode}[2002/04/16 v1.04]

\def\NormalCatcodes{%
  \endlinechar=13%
  \catcode33=12\catcode34=12\catcode35=6\catcode36=3\catcode37=14%
  \catcode38=4\catcode39=12\catcode40=12\catcode41=12\catcode42=12%
  \catcode43=12\catcode44=12\catcode45=12\catcode46=12\catcode47=12%
  \catcode58=12\catcode59=12\catcode60=12\catcode61=12\catcode62=12%
  \catcode63=12\catcode91=12\catcode92=0\catcode93=12\catcode94=7%
  \catcode95=8\catcode96=12\catcode123=1\catcode124=12\catcode125=2%
  \catcode126=13\catcode32=10\catcode13=5\catcode9=10\catcode10=12%
  \relax
}

\def\CatcodeStack{}
```

Notably absent from the following list: the `@` character and whitespace characters. The former is already handled in the LaTeX kernel, and I hesitate to interfere with its current catcode transitions.

```
\begingroup \escapechar=\m@ne \let\s\string
\xdef\pcat@otherchars{%
  \s\!\s\"\s\#\s\$\s\%\s\&\s\'\s\(\s\)\s\*\s +\s\,\s\-\s\.\s\/\s\:%
  \s\;\s\<\s\=\s\>\s\?\s\[\s\\\s\]\s\^\s\_\s\`\s\{\s\|\s\}\s\~%
}
\endgroup

\gdef\PushCatcodes{%
  \xdef\CatcodeStack{%
    \expandafter\PushCat@a\pcat@otherchars\ \
    \^^I\^^J{T \@gobbletwo}\@empty
    \relax
    {\CatcodeStack}}%
}
\def\PushCat@a#1{\catcode\number`#1=\number\catcode`#1 \PushCat@a}

\PushCatcodes \NormalCatcodes

\def\PopCatcodes{\expandafter\PopCat@a\CatcodeStack}
\def\PopCat@a#1#{#1\xdef\CatcodeStack}
```

2.2 Checking current settings

This can be used to store a copy of the current settings. Or print it with `\typeout`.

```
\def\CCSdo#1{ (\string#1\@iden{:\number\catcode`#1)\CCSdo}}
\def\CurrentCatcodesSubset{%
  \romannumeral 0\CCSdo\^^I\^^J\^^L\^^M\ \!\"\#\$\%\&\'\(\)\*+\,\-\.\/\:\;\<\=\>\?\@\[\\\]\^\_
}
```

2.3 Futurelet support

For code that does futureletting to see if a punctuation character follows, it is distressing to handle the case when some characters might be made active (e.g., by the babel package).

The `\FutureLetSetup` function attempts to minimize the difficulties by defining canonical control sequences to represent all the visible ASCII characters (i.e., ranging from space to `~`, 32–126, plus TAB) whose catcode could be changed without transgressing the limits of standard LaTeX syntax.

```
\begingroup\pcat@restore
\begingroup
\catcode`\3="3 \catcode`\4="4 \catcode`\7="7 \catcode`\8="8
\catcode`\A="A \catcode`\B="B \catcode`\C="C \catcode`\D="D
\gdef\fls@let#1#2{%
  \ifx\@@undefined#1\else\errmessage{\string#1 already defined}\fi
  \begingroup
  \lccode`\3=`#2\lccode`\7=`#2\lccode`\8=`#2%
  \lccode`\B=`#2\lccode`\C=`#2\lccode`\D=`#2\relax
  \ifnum\catcode`#2=\active
    \def\do##1{\noexpand\do\noexpand##1\noexpand}%
    \lowercase{%
      \xdef\fls@active@characters{%
```

```
        \fls@active@characters
        \do#1D}%
    }%
  \fi
  \aftergroup\global \aftergroup\let \aftergroup#1\aftergroup=%
  \lowercase{\aftergroup} %
  \lowercase{\expandafter\endgroup
    \ifcase\catcode`#2 %
      0\or \bgroup\or \egroup\or 3\or 4\or 5\or 6\or 7\or 8\or 9\or
      \@sptoken\or B\or C\else D\fi
  }%
}
\endgroup
```

`\dl@@dblquote` means "document-level double quote": a character token number 34 that has the catcode which is in effect at `\begin{document}` for double quotes.

```
\gdef\FutureLetSetup{%
  \gdef\fls@active@characters{}%
```

This will normally be the same as `\@sptoken`.

```
  \fls@let\dl@@space\ %
  \fls@let\dl@@exclam\!%
  \fls@let\dl@@dblquote\"%
  \fls@let\dl@@hash\#%
  \fls@let\dl@@dollar\$%
```

At document level, this cannot occur as a separate token.

```
%  \fls@let\dl@@percent\%%
  \fls@let\dl@@ampersand\&%
  \fls@let\dl@@rquote\'%
  \fls@let\dl@@lparen\(%
  \fls@let\dl@@rparen\)%
  \fls@let\dl@@star\*%
  \fls@let\dl@@plus\+%
  \fls@let\dl@@comma\,%
  \fls@let\dl@@hyphen\-%
  \fls@let\dl@@period\.%
  \fls@let\dl@@slash\/%
  \fls@let\dl@@colon\:%
  \fls@let\dl@@semicolon\;%
  \fls@let\dl@@less\<%
  \fls@let\dl@@equal\=%
  \fls@let\dl@@greater\>%
  \fls@let\dl@@question\?%
  \fls@let\dl@@lbracket\[%
```

At document level, this cannot occur as a separate token.

```
%  \fls@let\dl@@backslash\\%
  \fls@let\dl@@rbracket\]%
  \fls@let\dl@@hat\^%
```

```
    \fls@let\dl@@underscore\_%
    \fls@let\dl@@lquote\`%
```

Here one would normally use `\bgroup`.

```
    \fls@let\dl@@lbrace\{%
    \fls@let\dl@@vert\|%
```

Here one would normally use `\egroup`.

```
    \fls@let\dl@@rbrace\}%
    \fls@let\dl@@tilde\~%
  }
  \AtBeginDocument{\FutureLetSetup}

  \gdef\FutureLetReset{%
    \def\do##1##2{\let##1= ##2}%
    \fls@active@characters
    \let\do\relax
  }
  \endgroup

  \endinput \PopCatcodes\pcat@restore
```