This module implements some \special manipulation macros. I needed these when I implemented the code that handles the conversion of TPIC specials to PDF code.

When interpreting specials we need to do some basic scanning. For the moment we distinguish between three cases. We need
\special{tag: arguments}
\special{tag arguments}
\special{tag}

We cannot be sure that the first case isn’t
\special{tag: arguments}

So we have to take care of that one too.
\redefines.. Specials that are to be interpreted are defined with commands like:
\redefinespecial a: \using#1\endspecial%
{let’s execute special ‘a:’ using ‘#1’}
\redefinespecial a \using#1\endspecial%
{let’s execute special ‘a’ using ‘#1’}
\redefinespecial a \using#1\endspecial%
{let’s execute special ‘a’ using nothing}

The first two always take an argument, the last one not. The definition of this redefinition macro is not that complex. The names are internally tagged with \@rds@ which saves both time and space.

\mimmickspecials Mimmicking specials is activated by saying:
\mimmickspecials

This commands redefines the PLAIN \TeX primitive \special.

The special mimmicking macro first looks if it can find an colon terminated tag, next it searches for a tag that end with a space. If both cannot find, the tag itself is treated without argument.
Specials

8 \def\dodomimmickspecial#1 \relax/#4\end% 
   {\ifx#2\relax \dodomimmickspecial#1\using\endspecial 
      \else \dodomimmickspecial#1\using#2#3\endspecial \fi}

9 \def\dodomimmickspecial#1\using#2\endspecial% 
   {\expandafter\ifx\csname\@rds@#1\endcsname\relax % \doifdefelse 
      \defaultspecial#1 #2}% 
   \else \message{[mimmick special #1 with #2#3]}% 
       \getvalue{\@rds@#1}\using#2\endspecial \fi}

Now let’s show that things work the way we want, using the previous definitions of tag a.

\mimmickspecials
\special{a: 12345}
\special{a: 12345}
\special{a}

Which results in:

let’s execute special ‘a:’ using ‘12345’
let’s execute special ‘a:’ using ‘ 12345’
let’s execute special ‘a’ using nothing

When needed, one can call a mimicked special directly by saying for instance:

\mimmickspecial a: \using...\endspecial

This can be handy when specials have much in common.

10 \def\mimmickspecial #1 % 
   {\getvalue{\@rds@#1}}

Unknown specials are passed to the default special handler. One can for instance ignore all further specials by saying \normalspecial:

\def\defaultspecial#1{}

But here we default to idle.

11 \let\normalspecial =\special
   \let\defaultspecial=\special

12 \protect

13 \endinput
\texttt{\textbackslash defaultspecial 2} \hspace{1cm} \texttt{\textbackslash normalspecial 2}

\texttt{\textbackslash mimmickspecial 2} \hspace{1cm} \texttt{\textbackslash redefinespecial 1}

\texttt{\textbackslash mimmickspecials 1}