tokenizer.sty

Sascha Herpers∗
June 9, 2005

Abstract
This package provides a command \GetTokens, which splits a comma separated list of strings into tokens.\footnote{Herpers@wiso.uni-koeln.de}

Version: 1.1.0 (May 26, 2003)

1 Usage
This package defines the following command:
\GetTokens{<name1>}{<name2>}{<source>}.
\GetTokens can be used to split a comma separated list of strings passed as <source> into two tokens at the first encounter of a comma. While doing so, two commands with the name <name1> and <name2> are defined. The string up to the first comma of <source> is assigned as value for \<name1> and the rest is assigned to \<name2>.

By default the tokens are left as is, i.e. leading and trailing spaces are not removed. However, if this behavior is not desirable, you can use the package option trim. Specifying this option causes each token to be stripped of leading and trailing spaces.\footnote{Thanks to Harald Harders for his support.}

As a byproduct to the trim option the tokenizer package defines the following command:
\TrimSpaces{<source>}
\TrimSpaces can be used remove spaces from the text passed as first parameters.

\footnote{Omitting the trim options has the same result as using the notrim option.}
2 Example

The example shown in listing 1 illustrates the use of \texttt{\GetTokens}. Firstly, a source string \texttt{Source} is created, which contains the strings to be separated. The following while statement loops until there are no more tokens to process. \texttt{\GetTokens} is called and the separated tokens are stored in two commands \texttt{TokenOne} and \texttt{TokenTwo}, which are created by \texttt{\GetTokens}. Lastly, \texttt{Source} is replaced by the remainder string contained in \texttt{TokenTwo}.

\begin{verbatim}
\def\Source{ this , is , a , short , test }
The string \texttt{\emph{\Source}} contains the following tokens:\
\whiledo{\not\equal{\Source}{}}{{}
\GetTokens{TokenOne}{TokenTwo}{\Source}
\hspace*{.3cm}$\bullet$ \[
\texttt{TokenOne}\]
\let\Source\texttt{TokenTwo}
}
\end{verbatim}

Listing 1: example usage of \texttt{\GetTokens}

This is the output produced by the above example:

The string \texttt{this , is , a , short , test} contains the following tokens:
\begin{itemize}
\item \texttt{this}
\item \texttt{is}
\item \texttt{a}
\item \texttt{short}
\item \texttt{test}
\end{itemize}
3 History

<table>
<thead>
<tr>
<th>date</th>
<th>change</th>
</tr>
</thead>
<tbody>
<tr>
<td>05/26/03</td>
<td>added packages options <strong>trim</strong> and <strong>notrim</strong></td>
</tr>
</tbody>
</table>