`qolistings.sty` Package

Peter Brookes Chambers

November 18, 2020

Contents

1 Introduction

This package builds on the `listings` package to provide apealing and professional default styles with an easy interface for common options.

2 Useage

2.1 The `\qoinputlisting` Command

This command is analogous to the `\lstinputlisting` command provided by the `listings` package. It takes one mandatory argument, which should be the name of the file to include, and one optional argument which should be a series of comma separated key-value pairs (parsed by the `keyval` package). These key-value pairs are as follows, where the option is given on the left, the possible values in the centre (a "-" indicates that the option will default to "true" if no value is given), and the default value is given on the right.

2.2 Optional Arguments

The optional arguments for the commands in the `\qolistings` package are listed here in yellow, followed by the allowable values, then the default value. A value of "-" indicates that the key can be given with no values, in which case it is given a value of "true". The corresponding global options, where appropriate, are given in blue.

`lst options`	*any*	{}

This option takes as its argument a list of key-value pairs to be passed to the `listings` package, which should be encased in single braces only (for example, `lst options = {stringstyle = \color{red}, commentstyle = \tiny}`). Encasing these in double braces will cause errors as this will not be correctly expanded before being passed to the `keyval` package by `listings`.

These key-value pairs must also be valid options for a style in the `listings` package. These are applied **after** the default style (and any other options), and so if a key is given a value in both `lst options` and the default style, the value given in `lst options` is used.

`lst early options`	*any*	{}

This option is almost identical to the `lst options` option, except that it is applied **before** the default style (and subsequently before `lst options`), and so key-value pairs given in `lst early options` have the lowest priority. This doesn't currently have many direct applications, but should allow for more compatability should this package be expanded.

`latex comments`
`latexcomments`
	-, *true, false*	*false*

If true, then comments encased by the `escapeinside` tokens are rendered as normal LaTeX code rather than printed verbatim. By default, the escape tokens are "`(*`" and "`*)`", though these can of course be changed by passing `{escapeinside = {<token1>}{<token2>}}` to the `lst options` key.

`latex maths`	-, *true, false*	*false*
`latexmaths`		
`latex math`	-, *true, false*	*false*
`latexmath`		

If true (either with the U.S. or U.K. spelling), anything encased in dollar signs will be rendered as LaTeX maths rather than printed verbatim. Note that this applies to the entire file, not just to comments, and is independent of the `latex comments` option. This means that any $ anywhere in the file must be accompanied by a corresponding closing $, and the contents **must** be valid input for a LaTeX maths environment. (Be especially careful of escape sequences! Even strings written to be valid LaTeX when printed or passed to plotting tools may not be if backslashes need to be escaped.)

`latex`	-, *true, false*	
`latex`		

This flag simply sets both `latex comments` and `latex maths` to its own value. This is processed before the `latex comments` and `latex maths` options, so their values will overridde the `latex` flag.

`mdframed options`	*any*	{}

For better frame and background handling, all listings are encased in an `mdframed` environment. This option takes as its argument a list of key-value pairs to be passed to the `mdframed` package. Again, these should be encased in a single brace only.

These key-value pairs must also be valid options for a style in the `mdframed` package. These are applied last, after any other key-value pairs are handled for the `mdframed` environment.

`background`	-, *true, false*	*false*
`background`		

If true, the listing is given a background colour of `BackgroundColour`.

`rounded corners`	-, *true, false*	*false*
`roundedcorners`		

If true, the listing is given rounded corners with radius 0.5 em. This has no effect if both `background` and `frame` are false.

`frame`	-, *true, false, ltrb*	*false*
`frame`		

This option handles the frame of the listing. If true, then the listing is given a frame on all four sides. This option can also take any combination of “`l`”, “`t`”, “`r`”, and “`b`”, which correspond to a left, top, right, and bottom edge frame. Any combination of these (in any order) will draw a frame on each edge given. For example, `frame = blr` will draw a frame on all but the top edge.