The code in this project was written to illustrate a blog post and is not a full implementation of a paragraph "parser". It ignores a number of LuaTEX's node types that might be encountered during parsing of the paragraph because it is designed purely to illustrate concepts discussed in the blog post which accompanies this project. Note too that the parser code is not recursive: for example, if you detect an `hlist` or `vlist` during parsing then you would need to use recursive behavior to descend into those boxes. This is not difficult to implement but, again, for simplicity it is not done here.

Here is a typeset paragraph

For example, when TEX typesets a paragraph of text and breaks it into a series of lines, it considers the paragraph's text as a sequence of boxes and uses the dimensions of those character boxes to find the best linebreaks. Each line of the paragraph is itself a box (containing other boxes—e.g., characters) and the typeset paragraph lines (boxes) are stacked (vertically) to produce the paragraph. Eventually, the largest box of all is produced: the typeset page. Clearly, this is a very simplified picture because you also need the ability to arbitrarily position those boxes and TEX does this using so-called glue. Knuth commented (page 70 of The TEXbook) that "glue" probably should have been referred to as "spring" but the term glue was adopted early on and, to use Knuth's pun, it stuck.

Here is the same paragraph processed by a LuaTEX callback