

# richtext: Creating Rich Text Strings

D. P. Story  
Email: dpstory@acrotex.net

processed July 2, 2020

## Contents

<b>1 Introduction</b>	<b>1</b>
<b>2 Preliminaries</b>	<b>2</b>
<b>3 Documentation for the <b>RV</b> key</b>	<b>2</b>
3.1 The Font and Link tabs . . . . .	3
3.2 The Paragraph tab . . . . .	7
<b>4 Documentation for the <b>DS</b> key</b>	<b>11</b>
<b>5 Passing the rich content to <b>RV</b> and <b>V</b></b>	<b>13</b>
<b>6 Typesetting rich text mark up</b>	<b>14</b>
<b>7 Index</b>	<b>16</b>
<b>8 Change History</b>	<b>18</b>
1 <code>\*package</code>	
2 <code>\RequirePackage{xkeyval}</code>	
3 <code>\RequirePackage{ifpdf}</code>	
4 <code>\RequirePackage{ifxetex}[2006/08/21]</code>	
5 <code>\RequirePackage{eforms}</code>	
6 <code>\@ifundefined{ifpdfmarkup}{\newif{ifpdfmarkup}}{\pdfmarkupfalse}</code>	
7 <code>\ifpdf\else\ifxetex\else\pdfmarkuptrue\fi\fi</code>	

## 1 Introduction

This package supports the creation of *rich text strings* (a type of pdf string). A rich text string is used in a rich text field as the value of the PDF key **RV**. We also support the **DS** key which determines the default style.

From the PDF Reference (PDF 1.7), page 1310, “these rich text strings are fully-formed XML documents that conform to the rich text conventions specified

for the XML Forms Architecture (XFA) specification, which is itself a subset of the XHTML 1.0 specification, augmented with a restricted set of CSS2 style attributes.”

A rich text field may be created using the `eforms` package, like so

```
\textField[\Ff{\FfRichText}\Ff{\FfMultiline}\langle other-options\rangle
  \DS{\langle defaultstyle\rangle}\RV{\langle rich-value\rangle}\V{\langle plain-value\rangle}
]{\langle fldname\rangle}{\langle width\rangle}{\langle height\rangle}
```

This package provides commands and methods for ‘conveniently’ create values `\langle rich-value\rangle` and `\langle defaultstyle\rangle` for **RV** and **DS**; additionally, the value `\langle plain-value\rangle` of the **V** key is the ‘plain’ text value of the field; that is the text with all the formatting stripped out.

## 2 Preliminaries

```
8 \newif\ifrt@formfield \rt@formfieldtrue
9 \newif\ifrt@needsbody \rt@needsbodyfalse
10 \@ifpackageloaded{eforms}%
11   {\ifxetex\let\@eqV\@eqnuV\fi}{\rt@needsbodytrue}
12 \ifxetex\else\hypersetup{pdfencoding=pdfdoc}\fi
13 \providecommand\eq@RV@Body{<?xml version="1.0"?><body %
14   xfa:APIVersion="Acroform:2.7.0.0" %
15   xfa:contentType="text/html" %
16   xfa:spec="2.1" xmlns="http://www.w3.org/1999/xhtml" %
17   xmlns:xfa="http://www.xfa.org/schema/xfa-data/1.0/">}
18 \providecommand\eq@RV@endBody{</body>}
19 \def\rt@bBody{\ifrt@needsbody\eq@RV@Body\fi}
20 \def\rt@eBody{\ifrt@needsbody\eq@RV@endBody\fi}
```

## 3 Documentation for the RV key

We follow the Acrobat user interface. There two tabs of interest Font and paragraph.

### Font

#### Text

- Font: `<font-name>`
- Size: 10
- Baseline Shift: 0 points
- Underline: No Underline, Underline, Double Underline, Word Underline, Word Double Underline
- Style: Bold, Italic, Strike-through
- Color: RGB

### Paragraph

#### Alignment

- Horizontal: left, center, right, justify
- Vertical: top, middle, bottom

#### Indents

- Left: 0 points
- Right: 0 points
- First: None, First Line, Hanging
- By: 0 points

#### Spacing

- Above: 0 points
- Below: 0 points
- Line Spacing: Single, 1.5 Lines, Double Lines, Exactly (At: 0 points)

**Link** Enter a URL for this link

**On main Properties Menu bar:** Also supported are subscript (`<sub></sub>`) and superscript (`<sup></sup>`)

### 3.1 The Font and Link tabs

We support the attributes seen in the above list. We further support subscripts, superscript, and links in this section.

```
21 \newif\ifrtfontstyle\rtfontstylefalse
```

The keys of the `rtFont` key-value family. Supported keys are `font`, `size`, `raise`, `ulstyle`, `color`, `url`, and `style`. Superscripts and subscripts are handled differently.

**font** The `font` key's value is a font name, if the name contains a space, it should be enclosed in single quotes: `font=Arial` or `font='Myriad Pro'`.

```
22 \define@key{rtFont}{font}[]{\def\rt@argi{#1}\ifx\rt@argi\@empty
23 \let\rf@font\@empty\else\rtfontstyletrue
24 \def\rf@font{font-family:#1}\fi}
```

**size** The value of `size` is the size of the font, as measured in points `font=12pt`, note the use of the entity 'pt'.

```
25 \define@key{rtFont}{size}[]{\def\rt@argi{#1}\ifx\rt@argi\@empty
26 \let\rf@size\@empty\else\rtfontstyletrue\def\rf@size@num{#1}%
27 \def\rf@size@pt{#1pt}\def\rf@size{font-size:#1pt}\fi}
28 \def\rf@size@num{12}\def\rf@size@pt{12pt}
```

**raise** The key `raise` corresponds to the user interface item Baseline Shift, measured in points, for example, `raise=6pt`.

```
29 \define@key{rtFont}{raise}[]{\def\rt@argi{#1}\ifx\rt@argi\@empty
30 \let\rf@raise\@empty\else\rtfontstyletrue
31 \def\rf@raise{vertical-align:#1pt}\fi}
```

**ulstyle** The key `ulstyle` supplies an underline attribute, for example, `ul=word` underlines each word in the span.

```
32 \define@choicekey+{rtFont}{ulstyle}[\val\nr]%
33 {none,ul,2ul,wul,2wul}[none]{%
34 \ifcase\nr\relax
35   \def\rf@ul{none}\or
36   \def\rf@ul{underline}\or
37   \def\rf@ul{double}\or
38   \def\rf@ul{word}\or
39   \def\rf@ul{double word}\fi
40   \rtfontstyletrue
41 }{}
```

**color** The key `color` supplies a color attribute for the text in the span. There are two formats: `rrggbb` (hex) and `rgb(r,g,b)` (0-255). For example, `color=ff0000` or `color=rgb(255,0,0)` both color the text red.

```
42 \def\rt@r{r}\let\rt@One=1 \let\rt@Zero=0
43 \def\rt@parseColor#1(#2\@nil{\def\rt@argi{#2}\ifx\rt@argi\@empty
44 \let\rt@rgbdec\rt@Zero\else\let\rt@rgbdec\rt@One\fi}
45 \def\rt@gobbletonil#1\@nil{
46 \bgroup\@makeother\#\@makeother\&%
47 \gdef\rt@hashtag#\gdef\rt@amp{&}\egroup
48 \define@key{rtFont}{color}[]{\def\rt@argi{#1}\ifx\rt@argi\@empty
49 \let\rf@color\@empty\else\rtfontstyletrue
50 \rt@parseColor#1(\@nil
51 \if\rt@rgbdec\rt@One
52   \def\rf@color{color:#1}\else
53   \def\rf@color{color:\rt@hashtag#1}\fi
54 \fi}
55 \let\rf@color\@empty
```

**url** The key `url` enable the rich text string to contain a URL hypertext link.

```
56 \define@key{rtFont}{url}[]{\def\rt@argi{#1}\ifx\rt@argi\@empty
57 \let\rt@url\@empty\else\rtfontstyletrue\def\rt@url{#1}\fi}
```

**bold** The `rtFontStyle` family provides keys `bold`, `italic` and `strikeit`. They are possible values of the `style` key, define below. The `style` key can take on one or several values.

```
58 \define@choicekey+{rtFontStyle}{bold}[\val\nr]{normal,bold}[bold]%
59 {\edef\rfS@bold{\val}}{
60 \define@choicekey+{rtFontStyle}{italic}[\val\nr]{normal,italic}[italic]%
61 {\edef\rfS@italic{\val}}{
62 \define@key{rtFontStyle}{strikeit}[]{\def\rfS@strikeit{line-through}}
63 \let\rfS@normal\@empty\let\rfS@bold\@empty
64 \let\rfS@italic\@empty\let\rfS@strikeit\@empty
```

**style** Continuing the `rtFont` family, the `style` can take on several values: The key-value of `style={bold,italic,strikeit}` sets the text to bold, italic, and strike out. Multiple attributes must be enclosed in parentheses, as shown above.

```
65 \define@key{rtFont}{style}[]{\def\rt@argi{#1}\ifx\rt@argi\@empty
66 \let\rf@style\@empty\else\rtfontstyletrue\def\rf@style{#1}\fi}
```

`raw` We provide a `raw` experimental key. The value of this key is passed through; it must be of the proper syntax.

```

67 \define@key{rtFont}{raw}[]{\def\rt@argi{#1}\ifx\rt@argi\empty
68 \let\rf@raw\empty\else\rtfontstyletrue\def\rf@raw{#1}\fi}

```

`\resetRtFontKeys` The is an internal command to reset all keys to their default values.

```

69 \def\resetRtFontKeys{% rtFont family
70 \let\rf@font\empty\let\rf@size\empty
71 \let\rf@raise\empty\let\rf@ul\empty
72 \let\rf@color\empty\let\rf@style\empty\let\rt@url\empty
73 \let\rf@raw\empty
74 % rtFontStyle family
75 \let\rfS@normal\empty\let\rfS@bold\empty
76 \let\rfS@italic\empty\let\rfS@strikeit\empty
77 \rtfontstylefalse}

```

Now give all keys their default values.

```

78 \resetRtFontKeys

```

As we interpret the rich text string, we must save it properly formatted in both rich and plain format. These are macro for accumulating the strings.

```

79 \newcommand{\@AddToRichText}{\g@addto@macro\rt@RichText}
80 \newcommand{\@AddToPlainText}{\g@addto@macro\rt@PlainText}

```

Some utility commands

```

81 \def\rt@excl{!}
82 \def\rt@csarg#1#2{\expandafter#1\csname#2\endcsname}

```

`\useRV` The `\useRV` command expands to the rich string defined by `\rtpara` and is used as the value of the **RV** key.

`\useV` The `\useV` command expands to the plain string defined by `\rtpara` and is used as the value of the **V** key.

```

83 \newcommand{\useRV}[1]{\@nameuse{#1-ri}}
84 \newcommand{\useV}[1]{\@nameuse{#1-pl}}

```

`\rvorvstring` `\rvorvstring` is similar to `\texorpdfstring`, the first argument is a rich string while the second is a plain string. The two must match correctly, or the PDF reader may not display correctly; on error the reader displays the plain text.

```

85 \newif\if@rvstring \@rvstringfalse
86 \def\rvorvstring{\if@rvstring
87 \expandafter\@firstoftwo
88 \else
89 \expandafter\@secondoftwo
90 \fi
91 }

```

`\sub` **Subscripts (`\sub`) and superscripts (`\sup`)** There are two versions, one for rich text expansion and one for plain text expansion. Within `\rtpara` these two are `\let` to `\sub` and `\sup`.

```
92 \def\rt@sub#1{\rvorvstring{<sub>#1</sub>}{#1}}
93 \def\rt@sup#1{\rvorvstring{<sup>#1</sup>}{#1}}
```

`\br` Other supported markup: `\br`, `\bf`, and `\it`.

```
\bf 94 \def\rt@br{\rvorvstring{<br />}{\string\r}}
it 95 \def\rt@bf#1{\rvorvstring{<b>#1</b>}{#1}}
96 \def\rt@it#1{\rvorvstring{<i>#1</i>}{#1}}
97 \def\rt@spc{\rvorvstring{<span style="xfa-spacerun:yes">\rt&
98 \rt@hashtag160\rt@SC</span>}{}}
```

Some convenience commands

```
99 \def\rt@SC{;} \def\rt@CN{:} \def\rt@fs{font-style} \def\rt@fw{font-weight}
100 \def\rt@td{text-decoration}
```

**The `\span` command** There are two versions of the `\span` command, these are `\rt@remove@span` for plain text and `\rt@span` for rich text strings.

```
101 \def\rt@remove@span#1#2{#2}
```

`\rt@span` This is the internal `\span` command, it is `\let\span\rt@span`. Of course `\span` is a `TeX` primitive, so we must be careful not to overwrite it.

```
102 \def\rt@StyleAttr{\ifx\rf@font\@empty\else\rf@font\rt@SC\fi
103 \ifx\rf@size\@empty\else\rf@size\rt@SC\fi
104 \ifrt@formfield\ifx\rf@raise\@empty\else\rf@raise\rt@SC\fi\fi
105 \ifx\rf@ul\@empty\ifx\rfS@strikeit\@empty\else
106 \rt@td\rt@CN\rfS@strikeit\rt@SC\fi
107 \else\rt@td\rt@CN\rf@ul\ifx\rfS@strikeit\@empty\else\space
108 \rfS@strikeit\fi\rt@SC\fi
109 \ifx\rfS@bold\@empty\else\rt@fw\rt@CN\rfS@bold\rt@SC\fi
110 \ifx\rfS@italic\@empty\else\rt@fs\rt@CN\rfS@italic\rt@SC\fi
111 \ifx\rf@color\@empty\else\rf@color\rt@SC\fi
112 \ifx\rf@raw\@empty\else\rf@raw\fi}
```

`\span` The `\span` is let to `\rt@span` within the `\rtpara` command.

```
113 \let\rt@afterFont\relax
114 \newcommand\rt@span[2]{\resetRtFontKeys
115 \edef\x{\noexpand\setkeys{rtFont}{#1}}\x\rt@afterFont
116 \edef\x{\noexpand\setkeys{rtFontStyle}{\rf@style}}\x
117 \edef\rt@Style@ttr{\rt@StyleAttr}%
118 \ifx\rt@url\@empty
```

We are processing a regular `\span`.

```
119 \ifx\rt@StyleAttr\@empty\def\x{#2}\else
120 \edef\x{<span\ifrtfontstyle\space
121 style="\rt@Style@ttr"\fi>#2</span>}\fi
122 \else
```

We are processing a `\span` with the `url` key set.

```

123   \ifx\rt@StyleAttr@empty\edef\x{<a href="\rt@url">#2</a>}\else
124     \edef\x{<a href="\rt@url"
125       \ifrtfontstyle style="\rt@Style@ttr">#2</a>\fi}\fi
126   \fi
127 }
```

## 3.2 The Paragraph tab

We now come to the `\rtpara` command, which sets the attributes of the Paragraph tab.

`\rtpara` This is how you define a rich text string, through the use of `\rtpara`. The command takes three options: (1) The optional first takes key-values just defined in the `rtFont` and the `rtPara` families.

```

halign: text-align:left|center|right|justify
valign: text-valign:top|middle|bottom
        top is same as no text-valign attribute vertical-align
Indents:
  Left: margin-left:10pt;
  Right: margin-right:10pt
  First: text-indent: 12pt Indent
        text-indent:-12pt; Hanging
  None
Spacing > Line Spacing Line: height:18pt
Above margin-top:11pt; Below margin-bottom:11pt;
        applies to all text in field, not individual paragraphs
line-height: <num>pt
  Single Space: line-height:\rt@size
  1.5 Lines   : line-height: 1.8*max\rt@size
  Double      : line-height: 2.4*max\rt@size
  Exact       : line-height: <num>pt
```

`halign` **Alignment** The `halign` key effects the horizontal alignment of a paragraph, choices are `left`, `center`, `right`, and `justify`. The default is `left`.

```

128 \define@choicekey+{rtPara}{halign}[\val\nr]%
129   {left,center,right,justify}[left]{%
130   \ifcase\nr\relax
131     \def\rt@halign{text-align:left}\or
132     \def\rt@halign{text-align:center}\or
133     \def\rt@halign{text-align:right}\or
134     \def\rt@halign{text-align:justify}\fi
135   \rtfontstyletrue
136 }{}
```

`valign` The `valign` seems to effect all paragraphs in the rich text field. Its value deter-

mines the vertical placement of the paragraphs: `top`, `middle`, and `bottom`. The default is `top`.

```
137 \define@choicekey+{rtPara}{valign}[\val\nr]{top,middle,bottom}[top]{%
138   \ifcase\nr\relax
139     \def\rt@valign{text-valign:top}\or
140     \def\rt@valign{text-valign:middle}\or
141     \def\rt@valign{text-valign:bottom}\fi
142   \rtfontstyletrue
143 }{}
144 \let\rt@halign\@empty\let\rt@valign\@empty
```

`margleft` **Indents** With the `margleft` and `margright` you set the left and right margins of the effected paragraph. The default is `0pt`.

```
145 \define@key{rtPara}{margleft}{\def\rt@margleft{margin-left:#1pt}}
146 \define@key{rtPara}{margright}{\def\rt@margright{margin-right:#1pt}}
```

`indent` The `indent` key sets the amount of indent of a paragraph, values are `none`, `first`, and `hanging`. The amount of indent is determined by the key `indentby`, which is set to `12pt` by default.

`indentby`

```
147 \define@choicekey+{rtPara}{indent}[\val\nr]{none,first,hanging}[none]{%
148   \edef\rt@indenttype{\nr}%
149 }{}
150 \define@key{rtPara}{indentby}[12]{\def\rt@indentby{#1pt}}
151 \let\rt@margleft\@empty\let\rt@margright\@empty
152 \def\rt@indenttype{0}\def\rt@indentby{12pt}
```

`margtop` **Spacing** With the `margtop` and `margbottom` you set the space above and below a paragraph. The default is `0pt`.

`margbottom`

```
153 \define@key{rtPara}{margtop}[0]{\def\rt@margtop{#1pt}}
154 \define@key{rtPara}{margbottom}[0]{\def\rt@margbottom{#1pt}}
155 \def\rt@margtop{0pt}\def\rt@margbottom{0pt}
```

`linespacing` The `linespacing` key the spacing lines. The default is `0pt`. Choices are `single` (spacing), `oneandhalf` (spacing), `double` (spacing), and `exact` (spacing).

```
156 \define@choicekey{rtPara}{linespacing}[\val\nr]%
157   {single,oneandhalf,double,exact}[single]{%line-height
158   \edef\rt@linespacingtype{\nr}%
159   \ifcase\nr
160     \let\rt@linesp\@empty\or
161     \setlength{\@tempdima}{1.8pt*\rf@size@num}%
162     \edef\rt@linesp{\the\@tempdima}\or
163     \setlength{\@tempdima}{2.4pt*\rf@size@num}%
164     \edef\rt@linesp{\the\@tempdima}\or
165     \def\rt@linesp{\rf@size@pt}\fi
166 }{}
167 \let\rt@linesp\@empty\def\rt@linespacingtype{0}
```

`lineheight` The `lineheight` key

```
168 \define@key{rtPara}{lineheight}[]{\def\rt@lineheight{#1pt}}
```



```

169 \let\rt@lineheight\@empty
    More convenience definitions.
170 \def\rtpti{text-indent}\def\rtfmt{margin-top}
171 \def\rtpm{margin-bottom}\def\rtplh{line-height}
    We put the parameters all together.
172 \def\rt@ParaAttr{%
173 % Alignment
174 \ifx\rt@halign\@empty\else\rt@halign\rt@SC\fi
175 \ifrt@formfield
176 \ifx\rt@valign\@empty\else\rt@valign\rt@SC\fi
177 % Indents
178 \ifx\rt@margleft\@empty\else\rt@margleft\rt@SC\fi
179 \ifx\rt@margright\@empty\else\rt@margright\rt@SC\fi
180 \ifcase\rt@indenttype\space\or
181 \rtpti\rt@CN\rt@indentby\rt@SC\or
182 \rtpti\rt@CN-\rt@indentby\rt@SC\fi
183 % Spacing
184 \rtfmt\rt@CN\rt@margtop\rt@SC\rtpm\rt@CN\rt@margbottom\rt@SC
185 \ifx\rt@linesp\@empty\else
186 \if\rt@linespacingtype3%
187 \ifx\rt@lineheight\@empty
188 \rtplh\rt@CN\rt@linesp\rt@SC
189 \else
190 \rtplh\rt@CN\rt@lineheight\rt@SC
191 \fi
192 \else
193 \rtplh\rt@CN\rt@linesp\rt@SC
194 \fi
195 \fi
196 \fi }
197 \begingroup
198 \catcode'\@=0 \catcode'\@=12 \gdef\rtbs{\} \endgroup
199 \def\rt@cs#1{\rvorvstring{\rtbs\rtbs#1}{\string\134#1}}

```

**\rtpara** The `\tpara` takes three arguments. The first optional argument is key-value pairs from the `rtFont` and `rtPara` families. The second is a name this rich string. The third is the rich string itself, with supported markup.

```

200 \let\rt@afterParaFont\relax
    (2016/10/03) Added the dir HTML attribute, default is "ltr".
201 \def\rt@dir{ dir="ltr"}

```

We still have problems with `dvips` wrapping postscript lines around that break the code. Here we remove all `\spaces`'s with PDF spaces (`\040`), hopefully, there are no spaces at which to break the line and cause harm. `\pdfSP` and `\dl@sp@ce` are defined in `insdljs`.

```

202 \def\rt@sp@ce{ }
203 \def\rtpdfSPDef{\string\040}
204 \def\rtpdfSPDefPrnt{\string\040\allowbreak}

```

```

205 \bgroup\obeyspaces
206 \gdef\makePDFSp{\global\let =\pdfSP}%
207 \gdef\makeTeXSp{\global\let =\rt@sp@ce}%
208 \gdef\makeTeXSpPrnt{\global\let =\rtpdfSPDefPrnt}%
209 \egroup

```

Again, there are problems with EOL that disrupts RV key. Here, we define `\insertPDFSp@tEOL` to insert a PDF space (`\040`) at the end of the line. Seems to work.

```

210 \bgroup\catcode'\^^M=\active%
211 \gdef\insertPDFSp@tEOL{%
212   \catcode'\^^M=\active%
213   \let^^M\rtpdfSPDef%
214   \endlinechar='\^^M}%
215 \egroup

216 \newcommand\rtpara[2] []{\begingroup
217   \setkeys{rtPara,rtFont}{#1}\rt@afterParaFont
218   \edef\rt@Para@ttr{\rt@ParaAttr\rt@StyleAttr}%
219   \global\let\rt@RichText\@empty
220   \global\let\rt@PlainText\@empty
221   \def\rt@ctrlName{#2}%

```

Before taking the next parameter, we'll make some special definitions.

```

222   \def\{\string\}\def\}\{string\}}%
223   \def\1{\string\1}\def\2{\string\2}\def\3{\string\3}%
224   \@makeother\$\@makeother#\@makeother^\@makeother_\@makeother\~%
225   \@makeother\&\def\&{\rvorvstring{\string&}}{\string&}}% req
226   \@makeother\<\def\<{\rvorvstring{\string&lt;}}{\string<}}% req
227   \@makeother\>\def\>{\rvorvstring{\string&gt;}}{\string>}}%
228   \@makeother\' \def\' {\rvorvstring{\string&apos;}}{\string' }}%
229   \@makeother\" \def\" {\rvorvstring{\string&quot;}}{\string" }}%
230   \let\cs\rt@cs
231   \rtpara@cont}

```

`\rtpara` continues with `\rtpara@cont`. For pdfmarkup, we make special definitions that are not needed otherwise. `\makePDFSp` makes the space character into `\040`, `\insertPDFSp@tEOS` insert `\040` at the end of each line.

```

232 \def\rtpara@cont{\ifpdfmarkup
233   \makePDFSp\obeyspaces\insertPDFSp@tEOL\fi
234   \@ifnextchar\bgroup\rtpara@cont@i{\expandafter
235     \rtpara@cont@i@gobble}}

```

Now we take the last parameter, which is the rich text markup.

```

236 \def\rtpara@cont@i#1{%
237   \let\sup\rt@sup\let\sub\rt@sub
238   \let\br\rt@br\let\bf\rt@bf\let\it\rt@it
239   \let\spc\rt@spc
240   \let\span\rt@remove@span
241   \@rvstringfalse

```

Inserting a space at the beginning of plain text does no harm, but has benefits; it better matches the plain text with the rich context better.

```
242 \edef\x{#1}\expandafter\@AddToPlainText\expandafter{\x}%
243 \let\span\relax
244 \@rvstringtrue
245 \rtpara@i#1\span!;\endgroup}
```

Step 1: `\rtpara` comes here.

```
246 \def\rtpara@i#1\span#2;{\def\argii{#2}%
247 \@AddToRichText{#1}%
248 \ifx\argii\rt@excl
249 \rt@csarg\xdef{\rt@ctrlName-ri}%
250 {%
251 <p\rt@dir\ifx\rt@Para@ttr\@empty\else\space
252 style="\rt@Para@ttr"\fi>\rt@RichText</p>%
253 }%
254 \rt@csarg\xdef{\rt@ctrlName-pl}{\rt@PlainText}%
255 \let\rt@next\relax
256 \else
257 \def\rt@next{\rtpara@ii#2;}%
258 \fi
259 \rt@next}
```

Step 2: `\rtpara@i` comes here.

```
260 \def\rtpara@ii#1#2#3;{\def\argii{#2}%
261 \ifx\argii\rt@excl
262 \let\rt@next\relax
263 \else
264 \let\span\rt@span
265 \let\br\rt@br\let\bf\rt@bf\let\it\rt@it
```

We have encountered `\span{<argi>}{<argii>}` and we expand it appropriately.

```
266 \@rvstringtrue
267 \span{#1}{#2}\edef\rt@tmp{\noexpand
268 \@AddToRichText{\x}}\rt@tmp
269 \rt@csarg\xdef{\rt@ctrlName-ri}%
270 {%
271 <p\rt@dir\ifx\rt@Para@ttr\@empty\else\space
272 style="\rt@Para@ttr"\fi>\rt@RichText</p>%
273 }%
274 \rt@csarg\xdef{\rt@ctrlName-pl}{\rt@PlainText}%
275 \let\span\relax
276 \def\rt@next{\rtpara@i#3;}%
277 \fi\rt@next}
```

`skipline` There is a special definition for `skipline`, `skipline` is used between paragraphs to add a blank line between paragraphs.

```
278 \rt@csarg\xdef{\par-ri}{\rt@csarg\xdef{\par-pl}{\string\r}
279 \rt@csarg\xdef{\skipline-ri}%
280 {<p><span style="xfa-spacerun:yes">\rt@amp
281 \rt@hashtag160;</span></p>}
```

```

282 \rt@csarg\def{skipline-pl}{\string\r\space}
283 \def\rt@skipline{skipline}
284 \rt@csarg\def{br-ri}{}\rt@csarg\def{br-pl}{\string\r}

```

## 4 Documentation for the DS key

The **DS** key sets the default style. According to the JavaScript API for Acrobat reference, the default style supports alignment, textFont, (font family, font style, font weight), textColor, and textSize

```

/DSfont: Helvetica,sans-serif 12.0pt;text-align:left;color:#000000
f=this.getField"RichText"
style=f.defaultStyle;
style.fontFamily;
style.fontStyle;
style.fontWeight;
style.textFont;
style.alignment;
style.textColor;
style.textSize;

```

```

285 \def\rt@DSAttr{\ifx\rf@font\@empty\else\rf@font\rt@SC\fi
286 \ifx\rf@size\@empty\else\rf@size\rt@SC\fi
287 \ifx\rf@raise\@empty\else\rf@raise\rt@SC\fi
288 \ifx\rf@ul\@empty\ifx\rfS@strikeit\@empty\else
289 \rt@td\rt@CN\rfS@strikeit\rt@SC\fi
290 \else\rt@td\rt@CN\rf@ul\ifx\rfS@strikeit\@empty\else\space
291 \rfS@strikeit\fi\rt@SC\fi
292 \ifx\rfS@bold\@empty\else\rt@fw\rt@CN\rfS@bold\rt@SC\fi
293 \ifx\rfS@italic\@empty\else\rt@fs\rt@CN\rfS@italic\rt@SC\fi
294 \ifx\rf@color\@empty\else\rf@color\rt@SC\fi}

```

`\useDefaultDS` A fixed definition for default **DS**.

```

295 \newcommand\useDefaultDS{font-family:Helvetica,sans-serif;%
296 font-size:12.0pt;font-style:normal;font-weight:normal;%
297 text-align:left;color:\rt@hashtag000000}

```

`\setDefaultStyle` `\setDefaultStyle{myDS}{font=Arial,...,color=ff0000}`

```

298 \newcommand{\setDefaultStyle}[2]{\begingroup
299 \edef\x{\noexpand\setkeys{rtFont}{#2}}\x
300 \edef\x{\noexpand\setkeys{rtFontStyle}{\rf@style}}\x
301 \ifx\rf@ul\@empty\else
302 \let\rt@ul\@empty\PackageWarning{richtext}{%
303 The ul key is not supported within\MessageBreak
304 \string\setDefaultStyle. Ignoring it}\fi
305 \ifx\rf@raise\@empty\else
306 \let\rt@raise\@empty\PackageWarning{richtext}{%
307 The raise key is not supported within\MessageBreak

```

```

308     \string\setDefaultStyle. Ignoring it}\fi
309 \ifx\rt@url\@empty\else\let\rt@url\@empty
310   \PackageWarning{richtext}{%
311     The url key is not supported within\MessageBreak
312     \string\setDefaultStyle. Ignoring it}\fi

```

Fill in any missing essential attributes.

```

313 \ifx\rf@font\@empty
314   \def\rf@font{font-family:Helvetica,sans-serif}\fi
315 \ifx\rf@size\@empty\def\rf@size{font-size:\rf@sizept}\fi
316 \ifx\rf@color\@empty\def\rf@color{color:\rt@hashtag000000}\fi
317 \rt@csarg\xdef{#1-DS}{\rt@DSAttr}%
318 \endgroup

```

```

\useDS \useDS{<name>}
319 \newcommand{\useDS}[1]{\@nameuse{#1-DS}}

```

## 5 Passing the rich content to RV and V

```

\setRVVContent \setRVVContent{{name_1}{name_1}...{name_k}} or \setRVVContent{name}
320 \newif\ifrt@firsttok \rt@firsttoktrue
321 \newif\ifrt@itsskipline \rt@itsskiplinefalse

```

\rt@addtoRVV is a convenience internal command to add the name #1 both to \toks0, which holds the **RV** string, and to \toks2, which holds the **V** string.

```

322 \def\rt@addtoRVV#1{%
323   \toks4={\useRV{#1}}\edef\rt@tmpRV{\the\toks0\the\toks4}%
324   \toks4={\useV{#1}}\edef\rt@tmpV{\the\toks2\the\toks4}%
325   \toks0=\expandafter{\rt@tmpRV}\toks2=\expandafter{\rt@tmpV}%
326 }

```

We begin \setRVVContent

```

327 \def\rt@testifbgroup{\@ifnextchar\bgroup
328   {\let\rt@multiargs=1\rt@gobbletonil}
329   {\let\rt@multiargs=0\rt@gobbletonil}}
330 \newcommand{\setRVVContent}[2]{\begingroup
331   \rt@firsttoktrue \rt@itsskiplinefalse
332   \rt@testifbgroup#2\@nil
333   \def\contName{#1}\toks0={}\toks2={}\toks4={}%
334   \if\rt@multiargs1\def\rt@next{\setRVVContent@i#2;} \else
335     \def\rt@next{\setRVVContent@i{#2};}\fi\rt@next}
336 \def\setRVVContent@i#1{\def\rt@argi{#1}%
337   \ifx\rt@argi\rt@SC

```

If a semi-colon (\rt@SC), we are finished. Make the definitions for **RV** and **V**, and exit.

```

338   \rt@csarg\xdef{\contName-vcont}{\the\toks0}%
339   \rt@csarg\xdef{\contName-pcont}{\the\toks2}%
340   \let\rt@next\endgroup
341 \else

```

See if the current argument has been declared earlier by `\rtpara`. If not, we issue a warning and ignore it.

```

342   \expandafter\ifx\csname #1-ri\endcsname\relax
343     \PackageWarning{richtext}
344     {The name '#1' is not declared,\MessageBreak
345     will ignore it. Check the spelling}%
346     \def\rt@next{\setRVVContent@i}%
347   \else

```

We want to automatically induce `par` between non-`lineskip` tokens. The automatic `par` goes in prior to the token, so we first skip the first token.

```

348     \ifrt@firsttok\rt@firsttokfalse\else

```

Not the first token, see if it is a `skipline`, if yes, register it as a skip line for the next token.

```

349       \ifx\rt@argi\rt@skipline
350       \rt@itsskiplinetrue
351     \else

```

If the previous token was a `skipline`, we don't induce a `par`.

```

352       \ifrt@itsskipline
353       \rt@addtoRVV{br}\rt@itsskiplinefalse
354     \else

```

Finally, if this is not a `skipline`, and the previous token is not a `skipline`, we induce a `par`.

```

355       \rt@addtoRVV{par}%
356     \fi
357   \fi
358 \fi
359 \rt@addtoRVV{#1}%
360 \def\rt@next{\setRVVContent@i}%
361 \fi
362 \fi\rt@next
363 }

```

`\useRVContent{<name>}` Used to combine several paragraphs (**RV**)

```

364 \def\useRVContent#1{\@nameuse{#1-vcont}}

```

`\useVContent{<name>}` Used to combine several paragraphs (**V**)

```

365 \def\useVContent#1{\@nameuse{#1-pcont}}

```

## 6 Typesetting rich text mark up

It may be case that an author may want to display the underlying rich text markup and typeset it into the document for inspection and discussion. For this purpose, we offer

`displayRtPara{<name>}` Place an `\rtpara` command within and get a readout of **RV** and with `\displayRV{<name>}` and `\displayV{<name>}`.

```
366 \newenvironment{displayRtPara}[1]{%
367   \gdef\displayRtParaName{#1}\let\rtpdfSPDef\rt@sp@ce
368   \let\makePDFSp\makeTeXSp\let\rt@spc\rt@sp@ce
369   \def\rt@SC{;\allowbreak}\def\rt@CN{: \allowbreak}%
370 }{%
371   \rt@csarg\xdef{displayRV\displayRtParaName}%
372     {\useRV{\displayRtParaName}}
373   \rt@csarg\xdef{displayV\displayRtParaName}%
374     {\useV{\displayRtParaName}}
375 }
```

`displayRtPara*{<name>}` Place an `\rtpara` command within and get a readout of **RV** and with `\displayRV{<name>}` and `\displayV{<name>}`. In this version, spaces are displayed as `\040`, designed for the `dvips -> distiller` workflow. Prints the same result for all other workflows as `displayRtPara`.

```
376 \newenvironment{displayRtPara*}[1]{%
377   \gdef\displayRtParaName{#1}\let\rtpdfSPDef\rtpdfSPDefPrnt
378   \let\makePDFSp\makeTeXSpPrnt\let\rt@spc\rtpdfSPDefPrnt
379   \def\rt@SC{;\allowbreak}\def\rt@CN{: \allowbreak}%
380 }{%
381   \rt@csarg\xdef{displayRV\displayRtParaName}%
382     {\useRV{\displayRtParaName}}
383   \rt@csarg\xdef{displayV\displayRtParaName}%
384     {\useV{\displayRtParaName}}
385 }
```

`\displayRV{<name>}` displays the **RV** entry as defined by a `\rtpara` command with `<name>`.

`\displayV` Similarly, `\displayV{<name>}` displays the **V** entry

```
386 \def\displayRV#1{\csname displayRV#1\endcsname}
387 \def\displayV#1{\csname displayV#1\endcsname}
388 \</package>
```

## 7 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols		
<code>\#</code>	41, 219	<code>displayRtPara*</code> . . . . . 15
<code>\\$</code>	219	<code>\eq@RV@Body</code> . . . . . 8, 14
<code>\&amp;</code>	41, 220	<code>\eq@RV@endBody</code> . . . . . 13, 15
<code>\=</code>	193	
<code>\@AddToPlainText</code>	75, 237	<b>F</b>
<code>\@AddToRichText</code>	74, 242, 263	<code>font (key)</code> . . . . . 3
<code>\@eqV</code>	6	<b>G</b>
<code>\@eqnuV</code>	6	<code>\g@addto@macro</code> . . . . . 74, 75
<code>\@makeother</code>	41, 219–224	<b>H</b>
<code>\@rvstringfalse</code>	80, 236	<code>halign (key)</code> . . . . . 7
<code>\@rvstringtrue</code>	239, 261	<code>\hypersetup</code> . . . . . 7
<code>\@secondoftwo</code>	84	<b>I</b>
<code>\{</code>	217	<code>\if@rvstring</code> . . . . . 80, 81
<code>\}</code>	193, 217	<code>\ifpdf</code> . . . . . 2
<code>\^</code>	205, 207, 209, 219	<code>\ifpdfmarkup</code> . . . . . 2, 227
<code>\_</code>	219	<code>\ifrt@firsttok</code> . . . . . 315, 343
<code>\~</code>	219	<code>\ifrt@formfield</code> . . . . . 3, 99, 170
		<code>\ifrt@itsskipline</code> . . . . . 316, 347
<b>A</b>		<code>\ifrt@needsbody</code> . . . . . 4, 14, 15
<code>\active</code>	205, 207	<code>\ifrtfontstyle</code> . . . . . 16, 115, 120
<code>\allowbreak</code>	199, 364, 374	<code>\ifxetex</code> . . . . . 2, 6, 7
<code>\argii</code>	241, 243, 255, 256	<code>indent (key)</code> . . . . . 8
		<code>indentby (key)</code> . . . . . 8
<b>B</b>		<code>\insertPDFSp@tEOL</code> . . . . . 206, 228
<code>\bf</code>	6, 233, 260	<code>\it</code> . . . . . 6, 233, 260
<code>bold (key)</code>	4	<code>italic (key)</code> . . . . . 4
<code>\br</code>	6, 233, 260	<b>K</b>
		keys:
<b>C</b>		<code>bold</code> . . . . . 4
<code>color (key)</code>	4	<code>color</code> . . . . . 4
<code>\contName</code>	328, 333, 334	<code>font</code> . . . . . 3
		<code>halign</code> . . . . . 7
<b>D</b>		<code>indent</code> . . . . . 8
<code>\define@choicekey</code>	27, 53, 55, 123, 132, 142, 151	<code>indentby</code> . . . . . 8
<code>displayRtPara (environment)</code>	14	<code>italic</code> . . . . . 4
<code>displayRtPara* (environment)</code>	15	<code>lineheight</code> . . . . . 8
<code>\displayRtParaName</code>	362, 366–369, 372, 376–379	<code>linespacing</code> . . . . . 8
<code>\displayRV</code>	15, 381	<code>margbottom</code> . . . . . 8
<code>\displayV</code>	15, 382	<code>margleft</code> . . . . . 8
		<code>margright</code> . . . . . 8
<b>E</b>		<code>margtop</code> . . . . . 8
<code>\egroup</code>	42, 204, 210	
<code>\endlinechar</code>	209	
environments:		
<code>displayRtPara</code>	14	



raise	3	\rfs@italic	56, 59, 71, 105, 288
raw	4	\rfs@normal	58, 70
size	3	\rfs@strikeit	57, 59, 71, 100–103, 283–286
skipline	11	\rt@addtoRVV	317, 348, 350, 354
strikeit	4	\rt@afterFont	108, 110
style	4	\rt@afterParaFont	195, 212
ulstyle	3	\rt@amp	42, 92, 275
url	4	\rt@cargi	17, 20, 24, 38, 43, 51, 60, 62, 331, 332, 344
valign	7	\rt@bBody	14
		\rt@bf	90, 233, 260
		\rt@br	89, 233, 260
		\rt@CN	94, 101, 102, 104, 105, 176, 177, 179, 183, 185, 188, 284, 285, 287, 288, 364, 374
		\rt@cs	194, 225
		\rt@csarg	77, 244, 249, 264, 269, 273, 274, 277, 279, 312, 333, 334, 366, 368, 376, 378
		\rt@ctrlName	216, 244, 249, 264, 269
		\rt@dir	196, 246, 266
		\rt@DSAttr	280, 312
		\rt@eBody	15
		\rt@excl	76, 243, 256
		\rt@firsttokfalse	343
		\rt@firsttoktrue	315, 326
		\rt@formfieldtrue	3
		\rt@fs	94, 105, 288
		\rt@fw	94, 104, 287
		\rt@gobbletonil	40, 323, 324
		\rt@halign	126–129, 139, 169
		\rt@hashtag	42, 48, 93, 276, 292, 311
		\rt@indentby	145, 147, 176, 177
		\rt@indenttype	143, 147, 175
		\rt@it	91, 233, 260
		\rt@itsskiplinefalse	316, 326, 348
		\rt@itsskiplinetrue	345
		\rt@lineheight	163, 164, 182, 185
		\rt@linesp	155, 157, 159, 160, 162, 180, 183, 188
		\rt@linespacingtype	153, 162, 181
		\rt@margbottom	149, 150, 179
		\rt@margleft	140, 146, 173
		\rt@margright	141, 146, 174
		\rt@margtop	148, 150, 179
		\rt@needsbodyfalse	4
		\rt@needsbodytrue	6
		\rt@next	250, 252, 254, 257, 271, 272, 329, 330, 335, 341, 355, 357
		\rt@multiargs	323, 324, 329
		\rt@One	37, 39, 46
		\rt@Para@ttr	213, 246, 247, 266, 267
		\rt@ParaAttr	167, 213
		\rt@parseColor	38, 45
<b>L</b>			
lineheight (key)	8		
linespacing (key)	8		
<b>M</b>			
\makePDFSp	201, 228, 363, 373		
\makeTeXSp	202, 363		
\makeTeXSpPrnt	203, 373		
margbottom (key)	8		
margleft (key)	8		
margright (key)	8		
margtop (key)	8		
<b>O</b>			
\obeyspaces	200, 228		
<b>P</b>			
\PackageWarning	297, 301, 305, 338		
\pdfmarkupfalse	2		
\pdfmarkuptrue	2		
\pdfSP	201		
\providecommand	8, 13		
<b>R</b>			
\r	89, 273, 277, 279		
raise (key)	3		
raw (key)	4		
\RequirePackage	2		
\resetRtFontKeys	5, 64, 73, 109		
\rf@color	44, 47, 48, 50, 67, 106, 289, 311		
\rf@font	18, 19, 65, 97, 280, 308, 309		
\rf@raise	25, 26, 66, 99, 282, 300		
\rf@raw	63, 68, 107		
\rf@size	21, 22, 65, 98, 281, 310		
\rf@size@num	21, 23, 156, 158		
\rf@size@pt	22, 23, 160, 310		
\rf@style	61, 67, 111, 295		
\rf@ul	30–34, 66, 100, 102, 283, 285, 296		
\rf@url	304		
\rfs@bold	54, 58, 70, 104, 287		

<code>\rt@PlainText</code>	75, 215, 249, 269	<code>\rtpara@i</code>	240, 241, 271
<code>\rt@r</code>	37	<code>\rtpara@ii</code>	252, 255
<code>\rt@raise</code>	301	<code>\rtpdfSPDef</code>	198, 208, 362, 372
<code>\rt@remove@span</code>	96, 235	<code>\rtpdfSPDefPrnt</code>	199, 203, 372, 373
<code>\rt@rgbdec</code>	39, 46	<code>\rvorvstring</code>	5, 81, 87–92, 194, 220–224
<code>\rt@RichText</code>	74, 214, 247, 267		
<code>\rt@SC</code>	93, 94, 97–99, 101, 103– 106, 169, 171, 173, 174, 176, 177, 179, 183, 185, 188, 280–282, 284, 286–289, 332, 364, 374		
<code>\rt@skipline</code>	278, 344		
<code>\rt@sp@ce</code>	197, 202, 362, 363		
<code>\rt@span</code>	97, 259		
<code>\rt@spc</code>	92, 234, 363, 373		
<code>\rt@Style@ttr</code>	112, 116, 120		
<code>\rt@StyleAttr</code>	97, 112, 114, 118, 213		
<code>\rt@sub</code>	87, 232		
<code>\rt@sup</code>	88, 232		
<code>\rt@td</code>	95, 101, 102, 284, 285		
<code>\rt@testifbgroup</code>	322, 327		
<code>\rt@tmp</code>	262, 263		
<code>\rt@tmpRV</code>	318, 320		
<code>\rt@tmpV</code>	319, 320		
<code>\rt@ul</code>	297		
<code>\rt@url</code>	52, 67, 113, 118, 119, 304		
<code>\rt@valign</code>	134–136, 139, 171		
<code>\rt@Zero</code>	37, 39		
<code>\rtbs</code>	194		
<code>\rtfontstylefalse</code>	16, 72		
<code>\rtfontstyletrue</code>	18, 21, 25, 35, 44, 52, 61, 63, 130, 137		
<code>\rtplh</code>	166, 183, 185, 188		
<code>\rtplmb</code>	166, 179		
<code>\rtplmt</code>	165, 179		
<code>\rtplti</code>	165, 176, 177		
<code>\rtpara</code>	9, 123		
<code>\rtpara@cont</code>	226, 227		
<code>\rtpara@cont@i</code>	229–231		
		<b>S</b>	
		<code>\setDefaultStyle</code>	293
		<code>\setRVVContent</code>	315
		<code>\setRVVContent@i</code>	329–331, 341, 355
		<code>size (key)</code>	3
		<code>skipline (key)</code>	11
		<code>\span</code>	6, 235, 238, 240, 241, 259, 262, 270
		<code>\spc</code>	234
		<code>strikeit (key)</code>	4
		<code>style (key)</code>	4
		<code>\sub</code>	5, 232
		<code>\sup</code>	5, 232
		<b>T</b>	
		<code>\toks</code>	318–320, 328, 333, 334
		<b>U</b>	
		<code>ulstyle (key)</code>	3
		<code>url (key)</code>	4
		<code>\useDefaultDS</code>	290
		<code>\useDS</code>	314
		<code>\useRV</code>	78, 318, 367, 377
		<code>\useRVContent</code>	359
		<code>\useV</code>	78, 319, 369, 379
		<code>\useVContent</code>	360
		<b>V</b>	
		<code>valign (key)</code>	7
		<b>X</b>	
		<code>\x</code>	110, 111, 114, 115, 118, 119, 237, 263, 294, 295

## 8 Change History

v1.0.4 (2018/08/05)

`\rtpara`: Inserted space when adding to plain text . . . . . 10

v1.0.5 (2018/09/25)

`\rtpara`: Backing off that change . . . . . 10

v1.0a (2016/09/30)

`\setRVVContent`: Allow `\setRVVContent` to have only one argument. . . . . 13

v1.0c (2016/10/03)

`\rt@span`: Added `\rt@afterFont` to allow access by `annot_pro`. . . . . 6

`\rtpara`: Added `\rt@afterParaFont` to allow access by `annot_pro`. . . . . 9

Added `\rt@dir` . . . . . 9

v1.1 (2020/06/28)

General: Added `\displayRV` and `\displayV` . . . 15

Added `displayRtPara` and `displayRtPara*` . . 14

<code>\rtpara</code> : For pdfmarkup, spaces are now obeyed, and replaced with <code>\040</code> . . . . .	9	v1.1.1 (2020/07/02) General: Did not upload the docs, retry under new version number. . . . .	2
---	---	---	---