

# The `cuisine` package

Ben Cohen

2000/08/01, v0.5

## Abstract

This package provides an environment for typesetting recipes in steps in which each ingredient is on the left of the page next to the method step where it is used.

## 1 Introduction

There appear to be two styles of typesetting recipes in general use. The more common (at least in recipe books in the UK) is where the ingredients appear at the top above the method, like in the class `macros/latex/contrib/other/recipe`. Another way is to have each ingredient next to the method step in which it appears, as in the package `macros/latex/contrib/supported/cooking`; the `cuisine` package also uses this style, but closer to the format in *Practical Professional Cookery* by H. L. Cracknell and R. J. Kaufmann.

## 2 Usage

The package is loaded using the header `\usepackage{cuisine}`; the available options are:

- **number** Recipes will be numbered sequentially to the left of the recipe title. (This is the default.) The recipe *steps* will always be numbered.
- **nonumber** Recipes will not be numbered.
- **index** Recipe titles will be written to the contents for the document.
- **noindex** Recipe titles will not be written to the document contents. (This is the default.)

You will need the package `nicefrac`; it comes with current versions of `teTeX` and is also available from CTAN as part of the `units` package in `macros/latex/contrib/supported/units`.

**recipe** Each recipe is set out in a `recipe` environment. The three parameters are the title, and two recipe description fields—it is assumed that they are used for the number of servings and the preparation time, for example:

```

\begin{recipe}{Title} {Number of servings} {Preparation time}
...
\end{recipe}

```

There is no reason why the descriptive fields cannot be used for other things (such as the number of calories).

`\ingredient`      Within the environment, you type in method details directly, and you type  
`\ing`      the ingredients using `\ingredient` or the shorter version `\ing` before the method  
step beside which you want the ingredient to appear. A new method step is  
started whenever there is method text before an `\ingredient`. Two consecutive  
`\ingredient` declarations will appear next to the same method step.

The `\ingredient` command is used as follows:

```
\ingredient{Quantity} {Ingredient}
```

There is an optional first parameter which allows you to separate the numerical quantity from the name of the unit of measurement:

```
\ingredient[Numerical quantity] {Unit of measurement} {Ingredient}
```

Examples include:

```

\ingredient{a pinch}{salt}
\ingredient[4]{oz}{sugar}

```

`\fr`      Within the recipe you can use `\fr` to typeset fractions using `\fr12` to get  $\frac{1}{2}$ ,  
or `\fr{11}{12}` to typeset  $\frac{11}{12}$ . If the numerator or denominator of the fraction  
is more than one digit long then you will need to enclose it in braces otherwise  
they are optional.

`\degrees`      A degree symbol can be obtained using `\degrees` or the shorter form `\0`; for  
`\0`      example `120\0` gives  $120^\circ$ .

### 3 More advanced usage

The `cuisine` package has been designed so that it is easy to vary the widths of the columns.  $\text{\LaTeX}$  has very wide margins by default so the text width is very narrow which may be unsuitable for typesetting recipes using this package. (It might be worth looking at the KOMA-Script classes in `macros/latex/contrib/supported/koma-script/` as an alternative to the standard  $\text{\LaTeX}$  classes.)

`\RecipeWidths`      The `\RecipeWidths` command is designed to be used in conjunction with the  
 $\text{\LaTeX}$  commands for changing the page layout. It is used as follows:

```

\RecipeWidths{Total recipe width} {Recipe number width} {Number
of servings width} {Ingredient width} {Quantity width} {Units width}

```

<code>\ResetRecipeCounter</code>	The <code>\ResetRecipeCounter</code> command will reset the recipe counter so that subsequent recipes are numbered from 1.
<code>\newstep</code>	Within the recipe environment, to force a new step if you do not want to declare any ingredients, use the <code>\newstep</code> command. (A <code>\newstep</code> command with no preceding ingredient or method text will be ignored to prevent completely empty steps from appearing.)
<code>\newpage</code>	<code>\newpage</code> ends the current step (as for <code>\newstep</code> ) but also tells the environment that the next step will appear on the next page. Page breaks can only occur between method steps; method steps will not be split across pages. It is not normally necessary to use this command since page breaking after each step will be done automatically by L <sup>A</sup> T <sub>E</sub> X as necessary.
<code>\freeform</code>	To typeset text across the whole width of the recipe instead of just the width of the method, use the <code>\freeform</code> command. This is equivalent to <code>\noalign</code> in tables. Freeform text is not given a step number.

## 4 Fonts

The fonts used in the different parts of a recipe can be altered by changing the definition of the following macros; this should be done using, for example, `\renewcommand*{\recipetitlefont}{\sffamily}`.

<code>\recipefont</code>	The default font used for the fonts below
<code>\recipetitlefont</code>	The font for the recipe title
<code>\recipeservingsfont</code>	The font for the number of servings
<code>\recipetimefont</code>	The font for the preparation time
<code>\recipenumberfont</code>	The font for the recipe number
<code>\recipestepnumberfont</code>	The font for each step number
<code>\recipequantityfont</code>	The font for the numerical quantity
<code>\recipeunitfont</code>	The font for the unit of measurement (or descriptive quantity)
<code>\recipeingredientfont</code>	The font for the ingredient
<code>\recipefont</code>	The font for the method details
<code>\recipefreeformfont</code>	The font for freeform descriptions

## 5 Examples

### 5.1 A simple example

There is only whitespace between the first three ingredients, so they are used for the first step with the method text which follows. (Had there been text before the first ingredient, that text would become the first step and the heating instruction would be the second step.) The fourth `\ingredient` signals the end of the method text and starts a new step. The final `\end{recipe}` finishes that step.

```
\begin{recipe}{Yorkshire Pudding}{4 portions}{1\fr12 hours}
\ingredient[\fr12]{pt}{milk}
```

```

\ingredient[2]{oz}{butter}
\ingredient[5]{oz}{self-raising flour}
Heat the milk and butter until nearly boiling. Add flour and allow to
seeth over.
\ingredient[3]{}{eggs}
\ingredient{to taste}{salt and pepper}
Add the remaining eggs and whisk again. Cook at 220\0C for about 1 hour.
\end{recipe}

```

1	Yorkshire Pudding			4 portions
				1 <sup>1</sup> / <sub>2</sub> hours
1	1/2 pt	milk	Heat the milk and butter until nearly boiling. Add flour and allow to seeth over.	
	2 oz	butter		
	5 oz	self-raising flour		
2	3	eggs	Add the remaining eggs and whisk again. Cook at 220°C for about 1 hour.	
	to taste	salt and pepper		

## 5.2 A more complex example

We can change the widths of the columns using, for example,

```
\RecipeWidths{.75\textwidth}{0.5cm}{3cm}{1.75cm}{.75cm}{.75cm}
```

and the `small` environment to produce a different layout. We can also change the fonts, for example:

```

\renewcommand*{\recipetitlefont}{\large\bfseries\sffamily}
\renewcommand*{\recipequantityfont}{\sffamily\bfseries}
\renewcommand*{\recipeunitfont}{\sffamily}
\renewcommand*{\recipeingredientfont}{\sffamily}
\renewcommand*{\recipefreeformfont}{\itshape}

```

The example below demonstrates the `\newstep` and `\newpage` commands, and also how to use `\freeform` to create freeform text. `\freeform` can also be used to create rules in the same way as when using `\noalign` in tables. In this case, recipe numbering has been turned off, and the `small` environment is being used.

```

\begin{recipe}{Zabaglione alla Marsala}{4 Portions}{\fr12 hour}
\freeform This is a well-known Italian recipe which is
great for piling on the calories.
\ingredient[6]{}{egg yolks}
\ingredient[2]{oz}{granulated sugar}
\ingredient[6--8]{tbsp}{Marsala (or sherry)}
\ingredient[\fr14]{oz}{gelatine}
\ingredient[2]{tbsp}{cold water}

```

In the top of a double boiler, combine the egg yolks with sugar and Marsala, and whip the mixture over hot, but not boiling, water until it thickens. Stir in gelatine, softened in cold water and dissolved over hot water.

\ingredient[3]{tbsp}{brandy}

\ingredient[\fr38]{pt}{double cream}

Put the pan in a bowl of ice and stir the zabaglione well until it is thick and free of bubbles. When it is almost cold, fold in brandy and whipped cream and pour into individual moulds.

\freeform\rule{\textwidth}{0.05pt}\newpage

\freeform\rule{\textwidth}{0.05pt}

\ingredient[3]{}{egg yolks}

\ingredient[1]{oz}{granulated sugar}

\ingredient[3--4]{tbsp}{Marsala (or sherry)}

\ingredient[1\fr12]{tbsp}{brandy}

To make the sauce, combine egg yolks and sugar in the top of a double saucepan. Whisk mixture over hot, but not boiling, water until the sauce coats

the back of a spoon. Stir in the Marsala and brandy.

\newstep

Chill the zabaglione, unmould it, and serve with the sauce immediately.

\freeform\hrulefill

\end{recipe}

## Zabaglione alla Marsala

4 Portions

$\frac{1}{2}$  hour

*This is a well-known Italian recipe which is great for piling on the calories.*

1	<b>6</b>	egg yolks	In the top of a double boiler, combine the egg yolks with sugar and Marsala, and whip the mixture over hot, but not boiling, water until it thickens. Stir in gelatine, softened in cold water and dissolved over hot water.
	<b>2 oz</b>	granulated sugar	
	<b>6–8 tbsp</b>	Marsala (or sherry)	
	<b><math>\frac{1}{4}</math> oz</b>	gelatine	
	<b>2 tbsp</b>	cold water	
2	<b>3 tbsp</b>	brandy	Put the pan in a bowl of ice and stir the zabaglione well until it is thick and free of bubbles. When it is almost cold, fold in brandy and whipped cream and pour into individual moulds.
	<b><math>\frac{3}{8}</math> pt</b>	double cream	

---

3	<b>3</b>	egg yolks	To make the sauce, combine egg yolks and sugar in the top of a double saucepan. Whisk mixture over hot, but not boiling, water until the sauce coats the back of a spoon. Stir in the Marsala and brandy.
	<b>1 oz</b>	granulated sugar	
	<b>3–4</b> tbsp	Marsala (or sherry)	
	<b>1½</b> tbsp	brandy	
4			Chill the zabaglione, unmould it, and serve with the sauce immediately.

---

## 6 Bugs, Issues, Features, ...

- Vertical spacing of the boxes: the ingredients are not uniformly spaced and nor are the method steps. I currently have no idea how to solve this problem. (It is better than it was before I added a few extra `\struts` but if you compare the baselines of the ingredients to the adjacent method baselines, they are still not all properly aligned.)
- Vertical alignment of the recipe title: see `\r@cipetitle` on page 11.
- Support for dual sets of quantities, for example using a command like `\ingredient{1}{oz}{25}{g}{butter}`. This could be used where two systems of measurement are used or for different numbers of servings. The display would then have to have two extra columns or else (more likely) there would have to be an option to select which one to display.

Please e-mail me at [benc@cus.org.uk](mailto:benc@cus.org.uk) if you can help find or solve any problems with this package.

## 7 Identification section and initialisation

The package declaration follows the standard form.

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{cuisine}[2000/08/01 v0.5 recipe typesetting]
```

We use the `nicefrac` package for “nice” fractions.

```
3 \RequirePackage{nicefrac}
```

## 8 Counters, booleans and lengths

`r@cipenumber` is the recipe number counter. This could be made dependent on, say, the section number.

```
4 \newcounter{r@cipenumber}
```

`\ResetRecipeCounter` The `\ResetRecipeCounter` macro resets the recipe number counter to 0.

```
5 \DeclareRobustCommand{\ResetRecipeCounter}{%
6   \setcounter{r@cipenumber}{0}%
7 }
```

`st@pnumber` is the step counter within recipes. It is zeroed whenever the recipe counter is incremented.

```
8 \newcounter{st@pnumber}[r@cipenumber]
```

`ingr@dnumber` is the ingredient counter within recipes and steps. It is zeroed whenever the step counter is incremented.

```
9 \newcounter{ingr@dnumber}[st@pnumber]
```

`ifnumb@ring` determines whether or not recipe numbers are printed. (Step numbers will always be printed.)

```
10 \newif\ifnumb@ring
```

`ifind@xing` determines whether we are writing the recipes to an index file or not.

```
11 \newif\ifind@xing
```

`iffr@eforming` determines whether or not we are in freeform mode.

```
12 \newif\iffr@eforming
```

`ifn@wpaging` determines whether or not the next step will be on a new page.

```
13 \newif\ifn@wpaging
```

`\R@cipeWidth` is the total width of the recipe.

```
14 \newlength{\R@cipeWidth}
```

The following are, respectively, the widths of the recipe number box, the recipe title box, the recipe number-of-servings box, and the total of the recipe title and number-of-servings boxes.

```
15 \newlength{\R@cipeNumberWidth}
16 \newlength{\R@cipeTitleWidth}
17 \newlength{\R@cipeServingsWidth}
18 \newlength{\R@cipeTandSWidth}
```

This is the distance ingredient (etc.) paragraphs are outdented.

19 \newlength{\R@cipeOutdent}

These are the vertical distances for the recipe title-to-rule adjustments (usually negative).

20 \newlength{\R@cipeTitleVerticalAdjustTop}

21 \newlength{\R@cipeTitleVerticalAdjustBot}

These are the widths of the ingredient descriptors and combinations of them.

22 \newlength{\R@cipeIngredientWidth}

23 \newlength{\R@cipeQuantityWidth}

24 \newlength{\R@cipeUnitsWidth}

25 \newlength{\R@cipeQandUWidth}

26 \newlength{\R@cipeIandUWidth}

27 \newlength{\R@cipeIQUWidth}

\R@cipeMethodWidth is the width of the method text.

28 \newlength{\R@cipeMethodWidth}

\R@cipeStepWidth is the width of everything but the step number.

29 \newlength{\R@cipeStepWidth}

The following macros are to set the lengths defined above.

\R@cipeMethodWidths \R@cipeMethodWidths is to set the widths of the boxes/columns used in the recipe steps. There are three parameters: #1 is the ingredient width, #2 the quantity width, #3 the units width. The other lengths below are deduced from these and \R@cipeMethodWidth. The 0.5cm values are to leave a little space between the columns.

30 \DeclareRobustCommand\*\R@cipeMethodWidths[3]{%

31 \setlength{\R@cipeIngredientWidth}{#1}%

32 \setlength{\R@cipeQuantityWidth}{#2}%

33 \setlength{\R@cipeUnitsWidth}{#3}%

34 \setlength{\R@cipeQandUWidth}{\R@cipeQuantityWidth}%

35 \addtolength{\R@cipeQandUWidth}{\R@cipeUnitsWidth}%

36 \setlength{\R@cipeIQUWidth}{\R@cipeQandUWidth}%

37 \addtolength{\R@cipeIQUWidth}{\R@cipeIngredientWidth}%

38 \addtolength{\R@cipeIQUWidth}{0.5cm}%

39 \setlength{\R@cipeIandUWidth}{\R@cipeIQUWidth}%

40 \addtolength{\R@cipeIandUWidth}{-\R@cipeQuantityWidth}%

41 \setlength{\R@cipeStepWidth}{\R@cipeWidth}%

42 \addtolength{\R@cipeStepWidth}{-\R@cipeNumberWidth}%

43 \setlength{\R@cipeMethodWidth}{\R@cipeStepWidth}%

44 \addtolength{\R@cipeMethodWidth}{-\R@cipeIngredientWidth}%

45 \addtolength{\R@cipeMethodWidth}{-\R@cipeQandUWidth}%

46 \addtolength{\R@cipeMethodWidth}{-0.5cm}%

Allow for the ingredient paragraph outdenting.

47 \addtolength{\R@cipeIngredientWidth}{-\R@cipeOutdent}

48 \addtolength{\R@cipeUnitsWidth}{-\R@cipeOutdent}

49 \addtolength{\R@cipeQandUWidth}{-\R@cipeOutdent}

50 }%



`\R@cipeTitleWidths` `\R@cipeTitleWidths` is to set the widths of the boxes/columns used in the recipe title. There are two parameters: #1 is the recipe number width, #2 the number of servings width. The title width is deduced from there. The method widths are also updated with the new recipe number width.

```

51 \DeclareRobustCommand*\R@cipeTitleWidths}[2]{%
52   \setlength{\R@cipeNumberWidth}{#1}%
53   \setlength{\R@cipeServingsWidth}{#2}%
54   \setlength{\R@cipeTitleWidth}{\R@cipeWidth}%
55   \addtolength{\R@cipeTitleWidth}{-\R@cipeNumberWidth}%
56   \addtolength{\R@cipeTitleWidth}{-\R@cipeServingsWidth}%
57   \setlength{\R@cipeTandSWidth}{\R@cipeServingsWidth}%
58   \addtolength{\R@cipeTandSWidth}{\R@cipeTitleWidth}%
59   \R@cipeMethodWidths{\R@cipeIngredientWidth}{\R@cipeQuantityWidth}%
60                       {\R@cipeUnitsWidth}%
61 }

```

`\RecipeWidths` `\RecipeWidths` combines the above macros in a useable form. There are six parameters: #1 is the recipe width, #2 the recipe number width, #3 the number of servings width, #4 the ingredient width, #5 the quantity width, #6 the units width. The order of the three commands here is very important.

```

62 \DeclareRobustCommand*\RecipeWidths}[6]{%
63   \setlength{\R@cipeWidth}{#1}%
64   \R@cipeTitleWidths{#2}{#3}%
65   \R@cipeMethodWidths{#4}{#5}{#6}%
66 }%

```

Now we set up the default values. These appear to be reasonable for the LaTeX default page layout (A4, portrait, one column, etc.) I don't think the LaTeX default width is good for a recipe book though!

```

67 \setlength{\R@cipeOutdent}{0.3cm}%
68 \setlength{\R@cipeTitleVerticalAdjustTop}{-0.25cm}
69 \setlength{\R@cipeTitleVerticalAdjustBot}{-0.04cm}
70 \RecipeWidths{\textwidth}{0.8cm}{3cm}{3.5cm}{1cm}{1.7cm}

```

## 9 Fonts

This is where the default fonts are set up. `\recipefont` is the default font for the other recipe fonts; thus the other fonts inherit from this but can be altered individually.

```

71 \newcommand*\recipefont{\normalfont}
72 \newcommand*\recipetitlefont{\recipefont}
73 \newcommand*\recipenumberfont{\recipefont}
74 \newcommand*\recipestepnumberfont{\recipefont}
75 \newcommand*\recipequantityfont{\recipefont}
76 \newcommand*\recipeunitfont{\recipefont}
77 \newcommand*\recipeingredientfont{\recipefont}
78 \newcommand*\recipemethodfont{\recipefont}

```

```

79 \newcommand*\recipesservingsfont{\recipefont}
80 \newcommand*\recipetimefont{\recipefont}
81 \newcommand*\recipefreeformfont{\recipefont}

```

## 10 Boxes

`\st@pingrbox` is to hold the ingredients cumulatively as each `\ingredient` command is processed; `\st@pingrtmpbox` is used as a temporary transfer box.

```

82 \newsavebox{\st@pingrbox}
83 \newsavebox{\st@pingrtmpbox}

```

This is used to hold a single ingredient as it is being processed:

```

84 \newsavebox{\st@pIUbox}

```

This is to hold the method (or freeform) text for the step:

```

85 \newsavebox{\st@pmethodbox}

```

## 11 Options

**nonumber** — Should recipes be numbered? (The default is yes.)

```

86 \DeclareOption{number}{\num@ringtrue}
87 \DeclareOption{nonumber}{\num@ringfalse}

```

**index** — Should recipes be indexed to a file? (The default is no.)

```

88 \DeclareOption{index}{\ind@xingtrue}
89 \DeclareOption{noindex}{\ind@xingfalse}

```

Process the options, using the defaults if necessary.

```

90 \ExecuteOptions{number,noindex}
91 \ProcessOptions\relax

```

## 12 The main macros

### 12.1 Macros for displaying the steps and title

`\DisplaySt@p` `\DisplaySt@p` displays the ingredients and method for the current step and resets things ready for the next step.

```

92 \DeclareRobustCommand{\DisplaySt@p}{%

```

First, increment the step counter and put extra vertical space between steps. (But it isn't uniform yet, unfortunately.)

```

93 \stepcounter{st@pnumber}%

```

Display this step.

```

94 \makebox[\R@cipeWidth]{%
95   \makebox[\R@cipeNumberWidth][l]{\recipestepnumberfont\arabic{st@pnumber}}}%
96   \usebox{\st@pingrbox}%
97   \usebox{\st@pmethodbox}%
98 }%

```

Finally, clear the step storage boxes for the next step.

```

99 \savebox{\st@pingrbox}[\R@cipeIQUWidth]{}
100 }%

```

**\Fr@eFormStep** **\Fr@eFormStep** displays a free form description; it works in the same way as **\Displ@ySt@p**.

```

101 \DeclareRobustCommand{\Fr@eFormStep}{%
102 \usebox{\st@pmethodbox}%
103 \savebox{\st@pmethodbox}[\R@cipeMethodWidth]{}%
104 }%

```

**\DisplaySt@p** This calls one of the two display routines above, depending on whether we are freeforming or not.

```

105 \DeclareRobustCommand{\DisplaySt@p}{%
106 \iffre@forming%
107 \Fr@eFormStep%
108 \else%
109 \Displ@ySt@p%
110 \fi%
111 \ifn@wpaging%
112 \recipen@wpage%
113 \else%
114 \vskip0.2cm%
115 \fi%
116 \n@wpagingfalse%
117 }%

```

**\r@cipetitle** **\r@cipetitle** displays the recipe title construction, given the parameters #1 the title, #2 the number of servings and #3 the preparation time.  
Note: I want to get the recipe number aligned with the TOP of the title in the case of multi-line titles.

```

118 \DeclareRobustCommand{\r@cipetitle}[3]{
119
120 \bigskip
121 \pagebreak[0]

```

The following displays the text line above the rule:

```

122 \mbox{%
123 \ifnum@ring%
124 \makebox[\R@cipeNumberWidth][l]{\recipenumberfont\arabic{r@cipenumber}}%
125 \parbox[b]{\R@cipeTitleWidth}{\recipetitlefont #1}%
126 \else%
127 \parbox[b]{\R@cipeTitleWidth}{\recipetitlefont #1}%
128 \makebox[\R@cipeNumberWidth]{}%
129 \fi%
130 \parbox[b]{\R@cipeServingsWidth}{\hfill\recipeservingsfont #2}%
131 }\par%

```

Next we display the rule, being careful not to allow page breaks during the title, using the adjustments as specified above.

```

132 \nopagebreak
133 \vspace{\R@cipeTitleVerticalAdjustTop}%
134 \nopagebreak
135 \rule{\R@cipeWidth}{0.4pt}\par%
136 \nopagebreak
137 \vspace{\R@cipeTitleVerticalAdjustBot}%
138 \nopagebreak
    Finally we display the text line below the rule:
139 \makebox[\R@cipeWidth][r]{\recipetimefont #3}\par%
140 \nopagebreak
141 }%

```

## 12.2 Macros for the method box

These macros are to input the method box `\st@pmethodbox`, and are hence called at the start and end of the method box, or equivalently the *end* and *start* of the ingredient (etc.) macros.

```

\r@cipesloppy This is a version of which doesn't cause problems with the definition! (The LATEX
                definition has a problem where the empty box detection in \pr@ingred fails.)
142 \def\r@cipesloppy{%
143   \tolerance 9999%
144   \emergencystretch 3em%
145   \hfuzz.5pt%
146   \vfuzz.5pt%
    This is not in \sloppy: avoid warning on underfull boxes, which are more common
    since the columns are very narrow. This value may need to be adjusted...
147   \hbadness 1500%
148 }%

\pr@ingred This macro is called at the start of ingredients (that is, the end of the method box).
            It first ends the minipage-box which was started in the previous \p@stingred.
149 \DeclareRobustCommand{\pr@ingred}{%
150   \endminipage\end{lrbox}%
    Here we compare the width of the \st@pmethodbox with zero to determine whether
    anything was entered into it. (The idea for determining whether box is empty was
    from Donald Arseneau, comp.text.tex.)
151   \ifdim\wd\st@pmethodbox=0in%
152 %     \PackageWarning{cuisine}{No text}% % use for testing
153   \else%
154 %     \PackageWarning{cuisine}{Method text}%
155     \DisplaySt@p%
156   \fi%
157 }%

\p@stingred And this macro is called at the end of ingredients (the start of the method box).

```

This is very odd. Using `\begin{lrbox}...\end{lrbox}` isn't supposed to work properly over an environment definition (and doesn't seem to, either). So we use `\lrbox...\endlrbox`. But we want to nest a `\minipage` inside this, which fails. But it does work with `\begin{lrbox}\minipage...\endminipage\end{lrbox}`.

WHY?!

What happens, anyway, is that a new minipage-box is started. Provided the user doesn't put anything other than whitespace and comments between this and the next `\pr@ingred` then the box will be empty. This is how `\p@stingred` knows whether to display the step or not when called by `\ingr@dent`.

```
158 \DeclareRobustCommand{\p@stingred}{%
159   \fr@eformingfalse%
160   \begin{lrbox}{\st@pmethodbox}\minipage[t]{\R@cipeMethodWidth}%
161   \recipemethodfont%
162   \noindent%
163   \ignorespaces%
164   \r@cipesloppy%
165 }
```

## 12.3 Step terminating macros

These macros call `\pr@ingred` and `\p@stingred`, and can (through `\pr@ingred`) cause the next step to be started.

`\m@thodend` The `\m@thodend` command (which is `\newstep` in the environment) forces a new step unless there is no method **and** no ingredient.

```
166 \DeclareRobustCommand{\m@thodend}{%
167   \endminipage\end{lrbox}%
168   \ifdim\wd\st@pmethodbox=0in%
169     \ifnum\value{ingr@dnumber}>0%
```

In this case, the method box is empty so we put something in it to avoid problems with alignment.

```
170     \savebox{\st@pmethodbox}{\R@cipeMethodWidth}{\mbox{}}%
171     \DisplaySt@p%
172   \fi%
173   \else%
174     \DisplaySt@p%
175   \fi%
176   \p@stingred%
177 }
```

`\r@cipen@wpage` `\r@cipen@wpage` is a macro which sets `\n@wpagingtrue`—so that the next step will appear on the next page—and then forces the next step.

```
178 \DeclareRobustCommand\r@cipen@wpage{\global\n@wpagingtrue\m@thodend}
```

`\fr@eform` `\fr@eform` is for freeform text, like `\noalign` in tables. Again, as for `\m@thodend`, we want to force the new step unless there is no method and no ingredient.

```
179 \DeclareRobustCommand{\fr@eform}{%
```

```

180 \endminipage\end{lrbox}%
181 \ifdim\wd\st@pmethodbox=0in%
182   \ifnum\value{ingr@dnumber}>0%
183     \savebox{\st@pmethodbox}{\R@cipeMethodWidth}{\mbox{}}%
184     \DisplaySt@p%
185   \fi%
186 \else%
187   \DisplaySt@p%
188 \fi%

```

The rest is like \p@stingred:

```

189 \fr@eformingtrue%
190 \begin{lrbox}{\st@pmethodbox}\minipage[t]{\R@cipeWidth}%
191 \recipefreeformfont%
192 \noindent%
193 \ignorespaces%
194 \r@cipesloppy%
195 }%

```

`\ingr@dient` The `\ingr@dient` command takes 3 parameters, one optional. #3 is the ingredient. If #1 is non-blank then #1 is the numerical quantity and #2 the unit of measure, otherwise #2 is the quantity. Start by ending the method box.

```

196 \DeclareRobustCommand{\ingr@dient}[3][]{%
197   \pr@ingred%

```

The following is the main bit to typeset the ingredients list. (Thanks to Ulrike Fischer, `comp.text.tex`, for help with aligning box baselines.)

This is a hack to see if #1 is empty (Patrick Guio, `comp.text.tex`).

```

198 \ifx\relax#1\relax%

```

If it is empty we typeset #2 in the whole QandU width. We align the top of the ingredient box with the bottom of the quantity box but have the baseline as the top line of the quantity box. Use ragged right because the column is too narrow for flush right.

```

199   \savebox{\st@pIQUbox}{\R@cipeIQUWidth}[t]{%
200     \parbox[t]{\R@cipeIQUWidth}{%
201       \lineskipOpt\mbox{}\ll[-\baselineskip]%
202       \rule{\R@cipeOutdent}{0cm}%
203       \parbox[b]{\R@cipeQandUWidth}{%
204         \raggedright\recipeunitfont%
205         \setlength{\parindent}{-\R@cipeOutdent}%

```

This sees whether the text is large enough to fit on one line, in which case it centres it; otherwise it formats it sensibly.

Note: the `\hfill` matches another one elsewhere to centre the line.

```

206     \savebox{\st@pingrtmpbox}{#2}%
207     \ifdim\wd\st@pingrtmpbox>\R@cipeQandUWidth%
208       \rule{0pt}{\baselineskip}%
209       \strut #2\strut%
210     \else%

```

```

211         \noindent%
212         \rule{0pt}{\baselineskip}%
213         \strut #2\hfill\strut%
214     \fi%
215 }%
216 \rule{0.2cm}{0cm}%
217 \rule{\R@cipeOutdent}{0cm}%
218 \parbox[t]{\R@cipeIngredientWidth}{\raggedright%
219     \recipeingredientfont%
220     \setlength{\parindent}{-\R@cipeOutdent}%
221     \strut #3\strut}%
222 }%
223 }%
224 \else

```

If #1 was not empty, typeset #1 and #2 separately. First align the top of the ingredient box with the bottom of the unit box but have the baseline as the top line of the unit box; then repeat with that box below the quantity box.

```

225     \savebox{\st@pingrtmpbox}{\R@cipeIandUWidth}[t]{%
226         \parbox[t]{\R@cipeIandUWidth}{%
227             \lineskipOpt\mbox{}\[-\baselineskip]%
228             \rule{\R@cipeOutdent}{0cm}%
229             \parbox[b]{\R@cipeUnitsWidth}{%
230                 \raggedright\recipeunitfont%
231                 \setlength{\parindent}{-\R@cipeOutdent}%
232                 \rule{0pt}{\baselineskip}%
233                 \strut #2\strut\hfill}%
234             \rule{0.2cm}{0cm}%
235             \rule{\R@cipeOutdent}{0cm}%
236             \parbox[t]{\R@cipeIngredientWidth}{%
237                 \raggedright\recipeingredientfont%
238                 \setlength{\parindent}{-\R@cipeOutdent}%
239                 \strut #3\strut}%
240             }%
241         }%
242     \savebox{\st@pIQUbox}{\R@cipeIQUWidth}[t]{%
243         \parbox[t]{\R@cipeIQUWidth}{%
244             \lineskipOpt\mbox{}\[-\baselineskip]%
245             \parbox[b]{\R@cipeQuantityWidth}{\rule{0pt}{\baselineskip}%
246                 \hfill %
247                 \raggedright\recipequantityfont%
248                 \strut #1\strut%
249                 \rule{0.1cm}{0cm}}}%
250         \usebox{\st@pingrtmpbox}%
251     }%
252 }%
253 \fi%

```

Use \st@pingrtmpbox as a temporary holding box:

```

254 \savebox{\st@pingrtmpbox}{\R@cipeIQUWidth}[t]{\usebox{\st@pingrbox}}%

```

```

255 \savebox{\st@pingrbox}{\R@cipeIQUWidth}[t]{%
256   \begin{minipage}[t]{\R@cipeIQUWidth}%
257     \ifnum\value{ingr@dnumber}>0%
258       \usebox{\st@pingrtmpbox}\par%
259     \fi%
260     \usebox{\st@pIUbox}\strut%
261   \end{minipage}%
262 }%
263 % \usebox{\st@pingrbox} % For testing: show cumulative boxes.
264 \stepcounter{ingr@dnumber}%
265 \p@stingred%
266 }%

```

## 13 The recipe environment

**recipe** This is the main environment in the package. Its 3 parameters are #1 is the recipe title, #2 is the number of portions, and #3 the preparation time. Of course, #2 and #3 can be whatever you like but that is one way of using them.

```

267 \newenvironment{recipe}[3]{%

```

These are the things that are put at the start of the recipe environment. First, set things up. Increment the recipe counter, make command aliases and empty the boxes.

```

268   \stepcounter{r@cipenumber}
269   \let\newstep\m@thodend
270   \let\recipen@wpage\newpage
271   \let\newpage\r@cipen@wpage
272   \let\O\d@grees
273   \let\degrees\d@grees
274   \let\fr\fr@ction
275   \let\ing\ingr@dient
276   \let\ingredient\ingr@dient
277   \let\freeform\fr@eform
278   \n@wpagingfalse%
279   \setlength{\parindent}{0pt}
280   \savebox{\st@pingrbox}{\R@cipeIQUWidth}{ }
281   \savebox{\st@pmethodbox}{\R@cipeMethodWidth}{ }

```

Next deal with the index entry if we need to.

```

282   \ifind@xing
283     \addcontentsline{toc}{subsection}{#1}
284   \fi

```

Display the title and start the method box.

```

285   \r@cipetitle{#1}{#2}{#3}
286   \vskip0.2cm%
287   \p@stingred%
288 }%
289 {%

```



These are the things put at the end of recipes. End the method box and deal with the last step. Give a warning if the recipe is empty. It is apparently not necessary to reset the earlier assignments and `\parindent`.

```

290 \pr@ingred%
291 \ifnum\value{st@pnumber}=0% then complain!
292 \PackageWarning{cuisine}{The recipe did not have any steps}%
293 \fi%
294
295 \pagebreak[0]%
296 \medskip%

This prevents indentation problems after the end of the environment.
297 \@endpetrue%
298 }%
```

## 14 Miscellaneous useful macros

`\d@grees` This is to typeset a degrees symbol.

```

299 \DeclareRobustCommand{\d@grees}{%
300   ${}^{\circ}$%
301 }%
```

`\fr@ction` This is to typeset fractions, currently using package `nicefrac`.

```

302 \DeclareRobustCommand{\fr@ction}[2]{%
303   \nicefrac#1#2%
304 }%
```