User guide of *i*-LaTeX

Camille Gobert
gobert@lri.fr
April 2022

Contents

Quick reference of the special commands and environments

This list contains concise examples of all the special commands and environments that can be used with transitionals in *i*-LaTeX. Their purpose is explained in subsection 2.3, and they are presented in more details in section 3.

If you copy-paste code from here, do not forget to adapt the parameters as needed!

Mathematics

```
\begin{imaths} f(x) = 1 \end{imaths}
```
Insert a single- or multi-line formula

This environment has no parameter.
It is a wrapper around the `align*` environment (from the `amsmath` package).

Images

```
\iincludegraphics[width=\textwidth]{image.png}
```
Insert an image

This environment has two parameters: a list of optional key-value settings, followed by the path.
It is a wrapper around the `\includegraphics` command (from the `graphicx` package).

Tables

```
\begin{itabular}{ll} a & b \\ c & d \end{itabular}
```
Insert tabulated data

This environment has one parameter: the column types.
It is a wrapper around the `tabular` environment.

Grid layouts

```
\begin{gridlayout}{\textwidth}{8cm} ... \end{gridlayout}
```
Insert a grid layout

This environment has two parameters: the width of the grid, and the height of the grid.
It is a wrapper around the `minipage` environment.

```
\begin{row}{0.5} ... \end{row}
```
Insert a row (in a grid layout)

```
\begin{cell}{0.5} ... \end{cell}
```
Insert a cell (in a row)

These environments have one parameter: the relative size of the row/cell (between 0 and 1).
They are wrappers around `minipage` environments.

1 Introduction

i-LaTeX is a research prototype of a new kind of editor for LaTeX documents. It is built on top of Visual Studio Code (`https://code.visualstudio.com/`), an open-source code editor. *i*-LaTeX looks and works like other LaTeX editors such as TeXstudio and Overleaf (Figure 1), but it also offers a new kind of features we call *Transitional representations* (*Transitionals* for short).

Transitionals constitute an alternative way to visualise and manipulate certain parts of a LaTeX document than the source code or the generated PDF. Each transitional is bound to a single fragment of *visualisable code*, i.e. code that can be visualised through a transitional.

Transitionals are an optional feature of *i*-LaTeX: since the source code of your documents remains accessible at all times, you can also use *i*-LaTeX like a standard LaTeX editor and simply edit the code. While *i*-LaTeX may not always be able to parse and understand the code you write, in which case transitionals will be disabled, it will never prevent you from compiling a LaTeX document whose code is accepted by the LaTeX compiler itself.

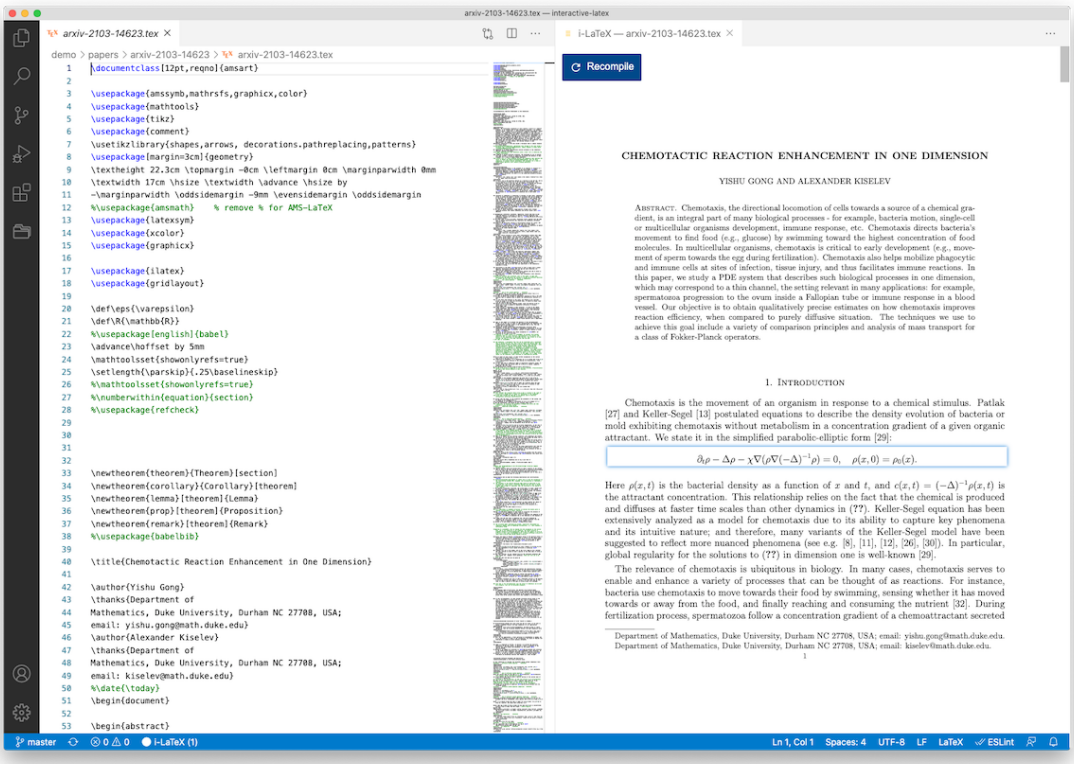


Figure 1: A typical layout to edit a LaTeX document opened with *i*-LaTeX: the code of the document is on the left, and the interactive PDF with transitionals is on the right.

2 General usage of *i*-LaTeX

2.1 How to setup the Visual Studio Code extension?

i-LaTeX is provided as an extension for the Visual Studio Code editor. It comes in the form of a single file with a `.vsix` extension. To install the extension from this file, you should open the extensions panel by clicking the Extensions button in the Activity bar (usually located on the left-hand side of the window). From there, you should click the icon with three dots in the top-right hand corner, select the "Install from VSIX..." entry, and select the `ilatex.vsix` file. The procedure is illustrated in Figure 2. Note that you can also uninstall the extension from this panel (by clicking on the gear icon next to the entry corresponding to the *i*-LaTeX extension in the list and selecting the "Uninstall" option).

Once the *i*-LaTeX extension in installed in Visual Studio Code, it will be automatically running in the background every time you open the editor. You can tell the *i*-LaTeX extension is activated by checking if there is an "i-LaTeX" button in the status bar (it should be displayed in the bottom-left hand corner of the window, as in Figure 3a).

When the extension is idle, the circle in the button is empty. On the contrary, when one or more interactive LaTeX documents are currently open, the circle in the button is filled, and the number of active documents is displayed between parentheses (Figure 3b).

2.2 How to open a LaTeX document with *i*-LaTeX?

In order to compile a LaTeX document with *i*-LaTeX and display the generated PDF, you must tell *i*-LaTeX which file is the *main* file of your document. This is the same file than the one Overleaf calls the *Main document*, as well as the file you would compile with a command-line tool such as `pdflatex` or `latexmk`). The simplest way to do this is to click the *i*-LaTeX button in the status bar: it will display a popup menu that contains different options to open a new LaTeX document with *i*-LaTeX (Figure 4). From this menu, you can either choose "Create from the active editor" to use the document open in the editor panel that currently has the focus or "Create from file…" to open a file selector. Every option has a variant that opens the file with *i*-LaTeX but disables transitionals (the PDF will still be displayed, but it will not be interactive).

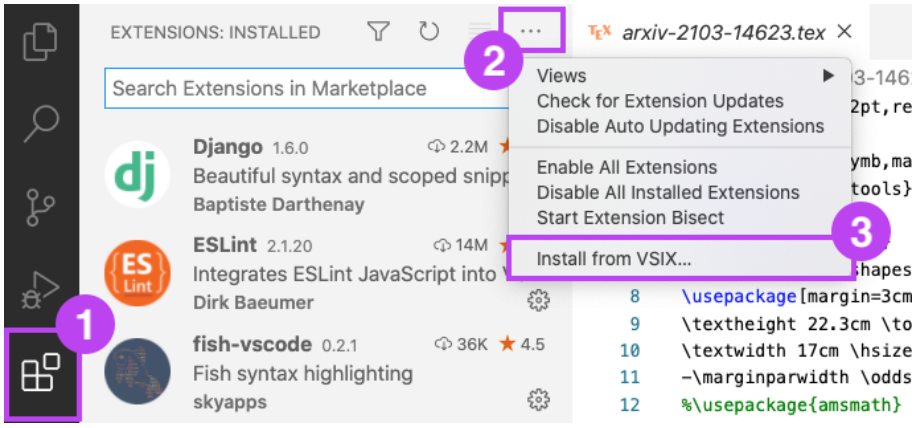


Figure 2: Procedure to install the *i*-LaTeX extension: ❶ open the Extensions panel; ❷ open the menu of the panel; ❸ select the VSIX file of the extension.

Finally, note that you can close an interactive document at any time by closing the panel that displays the PDF of that document.

(a) The left-hand side of the status bar.

(b) The two states of the *i*-LaTeX button.

Figure 3: The status bar of Visual Studio Code when the *i*-LaTeX extension is activated.

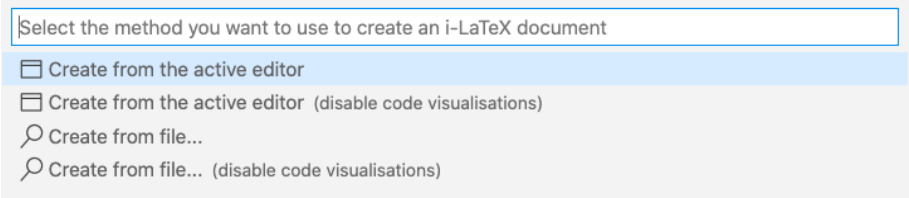


Figure 4: The popup menu that can be used to open a LaTeX document with *i*-LaTeX.

2.3 How to enable transitionals?

Every time you open a LaTeX document with *i*-LaTeX and every time you save a file, *i*-LaTeX will (1) recompile the LaTeX documents you opened with it and (2) attempt to locate every piece of visualisable code in your document and create a transitional for every one of them. Visualisable code is detected by the usage of **special LaTeX commands and environments** that are known to be visualisable by *i*-LaTeX. They are covered in more details in section 3.

In order to have access to these special commands and environments, you **must** use a special package named `ilatex`. To do so, you must put a copy of the `ilatex.sty` file in the same directory than the main file of every LaTeX document you would like to open with *i*-LaTeX, and you must include the package in the preamble of every document (i.e. before `\begin{document}`) using the command

```
\usepackage{ilatex}
```

Warning

When you edit the code of the document, *i*-LaTeX does its best to keep track of the start and the end of every piece of visualisable code in the document. However, in some situations, *i*-LaTeX may be **unable to understand** the syntax of your code, or it may **loose track** of a piece of visualisable code. In particular, the latter is likely to happen if you edit a selection that contains text that is both within and outside the range of a visualisable piece of code.

When it happens, *i*-LaTeX will show you it lost track of a piece of visualisable code by highlighting this piece of code in red. You have two options to fix this kind of problem:

- **In case of a syntax error** (e.g., a curly bracket "{" that has no matching closing bracket "}"), you can simply edit the code to fix the error, and *i*-LaTeX will remove the red highlighting as soon as the syntax becomes valid again.
- **Otherwise**, you should recompile the document to force *i*-LaTeX to re-detect all the visualisable pieces of code and re-create all the transitionals. This can be done by clicking the message displayed above the piece of code highlighted in red, by clicking the "Recompile" button displayed on top of the PDF, or by saving the document.

2.4 How to display transitionals?

Each piece of visualisable code is normally associated with an element in the PDF—the element that was generated by this piece of code. Every such element will be **surrounded by a blue halo** in the PDF displayed by *i*-LaTeX, and its transitional can be displayed by clicking on it. Depending on the position of the clicked element on your screen, the transitional will be displayed either below or above the element in the PDF, so that you can always look at the generated element while you use the transitional.

Some features are common to all transitionals:

- You can **show the code** that is being visualised by clicking on the title bar of the transitional (at the top of the popup). *i*-LaTeX will automatically open the file that contains the code (or give focus to the panel where it is currently open) and scroll to the appropriate position.
- You can **close the transitional** by either (1) clicking the button in the top-right hand corner or (2) clicking anywhere on the darkened background displayed on top of the PDF. If you modified the code of the document by interacting with the transitional, the document will automatically be **saved and recompiled** when the transitional is closed.

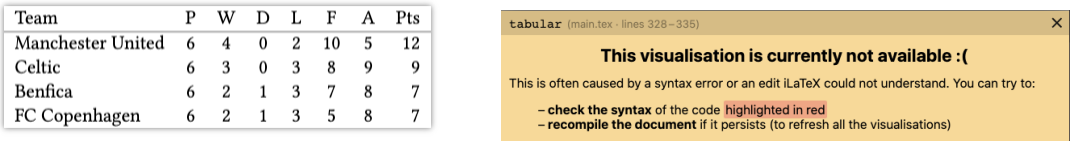
In some cases, the halo of one or several element may also turn grey (Figure 5a). A **grey halo** means that a transitional exists for this element but is **currently not available**.

The halo of an element will become grey if *i*-LaTeX is unable to parse the syntax of the code of this element. In addition, all halos will turn grey **every time your document is being compiled** (you cannot use a transitional while compiling), and **every time the last compilation of your document failed** (you will be notified by a popup message when it fails). If you fix all the errors and recompile your document, the halos should eventually turn blue again.

Good to know

When a transitional is displayed, you can of course interact with it, but you can also keep editing the source code of your document. *i*-LaTeX will automatically update the transitional to keep it synchronised with the piece of code it represents. You can use this mechanism to see how a certain change in the code affects the visualisation of the code provided by the transitional, without having to recompile the entire document.

In case you introduce a syntax error while editing the code, the transitional will turn yellow and display an error message (Figure 5b). The transitional should be restored as soon as the syntax becomes valid again. If a transitional is stuck in this state, try closing and reopening it. If it does not work, try saving the changes and recompiling the document.



Team	P	W	D	L	F	A	Pts
Manchester United	6	4	0	2	10	5	12
Celtic	6	3	0	3	8	9	9
Benfica	6	2	1	3	7	8	7
FC Copenhagen	6	2	1	3	5	8	7

(a) A PDF element with a grey halo.

(b) An error replacing a transitional.

Figure 5: (a) When a transitional becomes unavailable, the halo around the element it generated in the PDF becomes grey. (b) If the transitional is already displayed when becomming unavailable, the popup window turns yellow and displays an error message instead.

3 List of transitionals available in *i*-LaTeX

There are four kind of transitionals available in *i*-LaTeX. Each transitional is specialised for a particular kind of content, and each transitional is associated with a single command or environment. They are described in this section, along with their commands/environments.

Good to know

Special commands and environments cannot be nested, and is very likely to produce unexpected results if you try! However, the regular versions of the commands and environements can still be used as usual, including inside special commands and environements (e.g., using `tabular` inside `imaths` is valid).

3.1 Mathematics

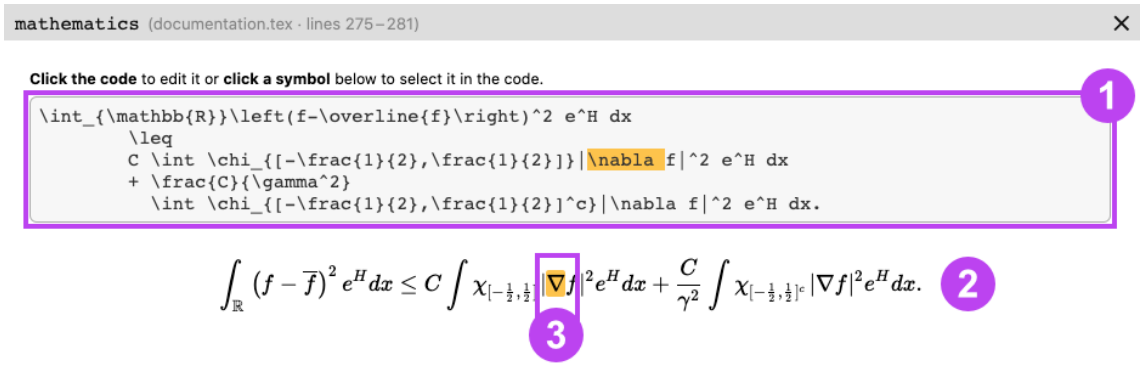


Figure 6: The transitional for `imaths` environments. 1 The editable code of the formula. You can click it to **modify the code**, and the typeset formula will be **updated in real time**. 2 A typeset formula representing the code displayed above. 3 You can **point** and **click** most symbols in the typeset formula to respectively **highlight** or **select the piece of code** that generated it.

A transitional for mathematical formulae can be created with the `imaths` environment (Figure 6):

```
\begin{imaths}
    x = \alpha x + \beta y + \gamma
\end{imaths}
```

It behaves like the `align*` environment provided by the `amsmath` package: you can use all the commands accepted by LaTeX in math mode inside, and you can align several formulae by using `&` to align symbols vertically and `\\` to break lines.

Good to know

When you edit the code of the formula in the transitional, the actual code of your LaTeX document will **not** be updated until you click outside of the text area or press Enter.

Warning

Some uncommon commands or environments may not be supported (it will compile, but the transitional will not work). User-defined commands are not supported either, unless the commands are defined *within* the `imaths` environment.

Example

Writing

```
\begin{imaths}
    \int_{\mathbb{R}}\left(f-\overline{f}\right)^2 e^H dx
    \leq
    C \int \chi_{[-\frac{1}{2},\frac{1}{2}]}|\nabla f|^2 e^H dx
    + \frac{C}{\gamma^2}
      \int \chi_{[-\frac{1}{2},\frac{1}{2}]^c}|\nabla f|^2 e^H dx.
\end{imaths}
```

will produce

$$\int_{\mathbb{R}} \left(f-\overline{f}\right)^2 e^H dx \leq C \int \chi_{[-\frac{1}{2},\frac{1}{2}]}|\nabla f|^2 e^H dx + \frac{C}{\gamma^2} \int \chi_{[-\frac{1}{2},\frac{1}{2}]^c}|\nabla f|^2 e^H dx.$$

3.2 Images

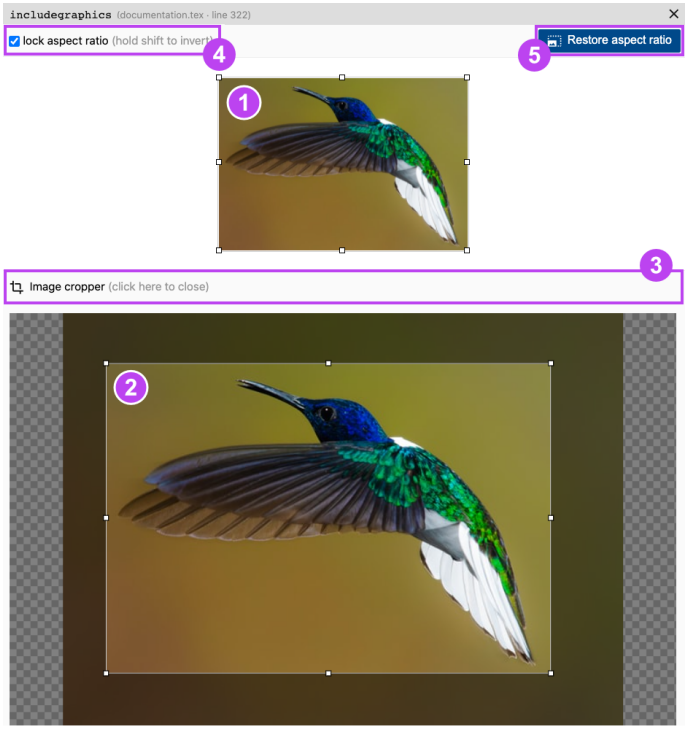


Figure 7: The transitional for `iincludegraphics` commands. 1 The expected output of the command. You can **resize it** by dragging one of the handles around the image. 2 The area of the image to display. You can **move it** by dragging it and **resize it** by dragging one of the handles around the visible area. 3 You can display or hide the image cropper by clicking this bar. 4 You can hold the Shift key or uncheck this box to resize the image with a **non-natural aspect ratio**. If the image has a non-natural aspect ratio when the transitional is opened, the box will be unchecked by default. 5 You can resize the image to **restore its natural aspect ratio** by clicking this button.

A transitional for images can be created with the `\iincludegraphics` command, with two `is` (Figure 7):

```
\iincludegraphics[width=\textwidth]{path/to/your/image.png}
```

It behaves like the `\includegraphics` command provided by the `graphicx` package. Only the `width`, `height`, `trim` and `clip` options are recognised by *i*-LaTeX. You can still use all the other options supported by `\includegraphics`, but they will be ignored and may be overwritten by the transitional when you interact with it.

Good to know

i-LaTeX supports the most common *length macros* like `\textwidth`, so you can freely use them in the options of a command like this one. It also supports the relative length units `em` and `ex`.

Warning

PDF images are not supported (it will compile, but the transitional will not work correctly).

Example

Writing

```
\iincludegraphics[width = 0.5\textwidth]{img/bird.jpg}
```

will produce

3.3 Tables

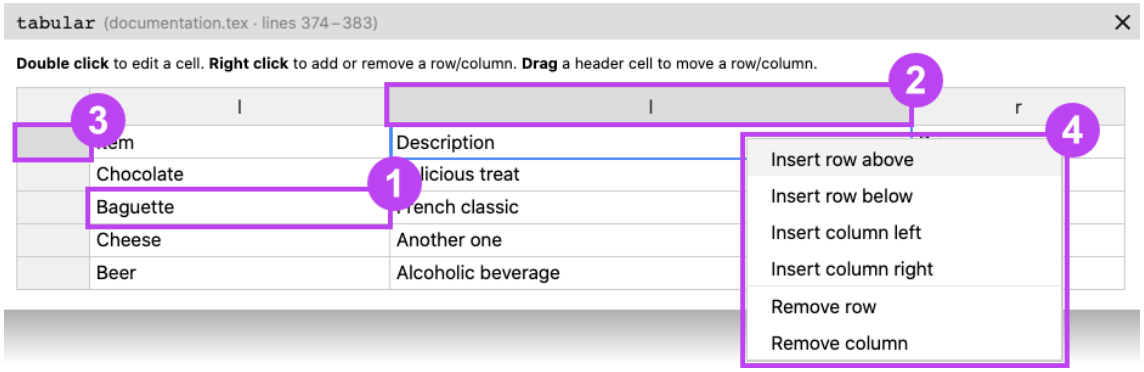


Figure 8: The transitional for `itabular` environments. 1 A cell of the table. You can **edit** its content by double-clicking on it. 2 The header cell of a column. If *i*-LaTeX was able to parse the type of this column (e.g., `l`, `c`, `r`), this cell contains it. You can **drag it** to **move the column**. 3 The header cell of a row. You can **drag it** to **move the row**. 4 You can display a **contextual menu** by right-clicking any cell of the table. It enables you to **insert** and **delete rows and columns**.

A transitional for tables can be created with the `itabular` environment (Figure 8):

```
\begin{itabular}{llr}
    Item      & Description & Price \\
    Chocolate & ...         & 2.20  \\
    Baguette  & ...         & 0.90
\end{itabular}
```

It behaves like the `tabular` environment, and expects a mandatory argument (the column types).

Good to know

Common formatting commands for tables such as `\hline`, `\toprule`, `\midrule` and `\bottomrule` are not considered as "cell content" by this transitional. You can safely use them, and *i*-LaTeX will do its best to ignore them and leave them in place.

Warning

This transitional expects every line to contain the same number of cells, and is likely to dysfunction if it is not the case. In particular, that means that merging cells with commands such as `\multicolumn` and `\multirow` is not supported (it will compile, but the transitional will not work correctly).

Example

Writing

```
\begin{itabular}{llr}
    \toprule
    Item & Description & Price \\
    \midrule
    Chocolate & Delicious treat & 2.20  \\
    Baguette & French classic & 0.90  \\
    Cheese & Another one & 3.00  \\
    Beer & Alcoholic beverage & 4.50  \\
    \bottomrule
\end{itabular}
```

will produce

Item	Description	Price
Chocolate	Delicious treat	2.20
Baguette	French classic	0.90
Cheese	Another one	3.00
Beer	Alcoholic beverage	4.50

3.4 Grid layouts

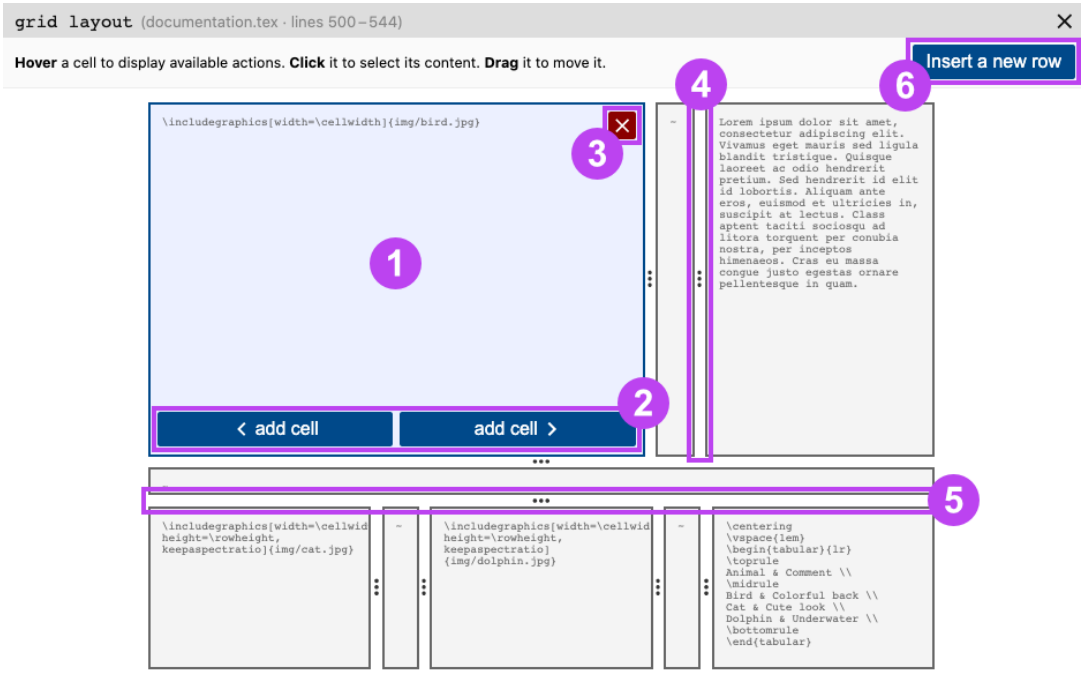


Figure 9: The transitional for `gridlayout` environments. ❶ A cell of the layout that is pointed by the mouse. It can be dragged to be moved before or after another cell. ❷ Buttons to insert a new cell. ❸ Button to remove the cell. ❹ Separator between two cells. It can be dragged to resize the adjacent cells. ❺ Separator between two rows. It can be dragged to resize the adjacent rows. ❻ Button to insert a new row at the bottom of the grid.

A transitional for grid layouts can be created with the `gridlayout` environment (Figure 9):

```
% This grid is as wide as the text (\textwidth) and 8cm tall
\begin{gridlayout}{\textwidth}{8cm}
    \begin{row}{0.6}
        \begin{cell}{1}
            % A cell as wide as the row that contains it
        \end{cell}
    \end{row}
    \begin{row}{0.4}
        \begin{cell}{0.33}
            % Bottom-left cell taking 1/3 of the row
        \end{cell}
        \begin{cell}{0.67}
            % Bottom-right cell taking 2/3 of the row
        \end{cell}
    \end{row}
\end{gridlayout}
```

Contrary to the other transitionals, this transitional is not simply a wrapper around an existing command or environment, but an **experimental structure** built to explore how an editor such as *i*-LaTeX could make it easier to manipulate this type of layout in LaTeX. Internally, it uses nested `minipage` environements. Every grid can contain rows (`row` environments), and every

row can contain cells (`cell` environments). A cell can contain **arbitrary content** (text, image, table, etc), but it **cannot contain other special commands or environments** such as `\iincludegraphics`.

The `grid` environment expects two mandatory arguments: the **width** and the **height** of the grid. The `row` and `cell` environments expect one mandatory argument each: the **relative height** of the row and the **relative width** of the cell. These relative dimensions are **unitless numbers between 0 and 1** that must sum to 1 over all the cells of a row and over all the rows of the grid. If they do not sum to 1, the document will compile, but the cells and the rows may not be positioned correctly, and the transitional may not work correctly.

Good to know

In every cell, you can use `\cellwidth` and `\rowheight`, two custom length macros relative to the current cell (e.g., to make an image have the same width or height than the cell).

Warning

You should always use **at least one row** with **at least one cell** in a `gridlayout` environment, and you should avoid putting anything outside of a `cell` environment. Otherwise, you are likely to break the layout, and the transitional will not represent it correctly anymore.

Warning

Empty cells, such as cells acting as white space, appear to be ignored when the `cell` environments contain no symbol. You can fix this issue by putting a whitespace character in the body of the environment, such as a non-breaking space (`~`).

Example

Writing

```
\begin{gridlayout}{\textwidth}{11cm}
    \begin{row}{0.65}
        \begin{cell}{0.65}
            \includegraphics[width=\cellwidth]{img/bird.jpg}
        \end{cell}
        \begin{cell}{0.05}
            ~ % Separator
        \end{cell}
        \begin{cell}{0.3}
            Lorem ipsum dolor sit amet [...] in quam.
        \end{cell}
    \end{row}
    \begin{row}{0.05}
```

```
        \begin{cell}{1}
            ~ % Separator
        \end{cell}
    \end{row}
    \begin{row}{0.3}
        \begin{cell}{0.3}
            \includegraphics[
                width=\cellwidth,
                height=\rowheight,
                keepaspectratio
            ]{img/cat.jpg}
        \end{cell}
        \begin{cell}{0.05}
            ~ % Separator
        \end{cell}
        \begin{cell}{0.3}
            \includegraphics[
                width=\cellwidth,
                height=\rowheight,
                keepaspectratio
            ]{img/dolphin.jpg}
        \end{cell}
        \begin{cell}{0.05}
            ~ % Separator
        \end{cell}
        \begin{cell}{0.3}
            \centering
            \vspace{1em}
            \begin{tabular}{lr}
                \toprule
                Animal & Comment \\
                \midrule
                Bird & Colourful back \\
                Cat & Cute look \\
                Dolphin & Underwater \\
                \bottomrule
            \end{tabular}
        \end{cell}
    \end{row}
\end{gridlayout}
```

will produce (scaled to fit here)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus eget mauris sed ligula blandit tristique. Quisque laoreet ac odio hendrerit pretium. Sed hendrerit id elit id lobortis. Aliquam ante eros, euismod et ultricies in, suscipit at lectus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Cras eu massa congue justo egestas ornare pellentesque in quam.

Animal	Comment
Bird	Colourful back
Cat	Cute look
Dolphin	Underwater