The texpower Package

Creating dynamic online presentations with LaTeX

Stephan Lehmke

mailto:Stephan.Lehmke@cs.uni-dortmund.de

March 22, 2000

This document is a demonstration and preliminary manual for the texpower package which allows to create incremental presentations in a very flexible way.

This package implements some commands for creating presentations with the combination of LaTeX + dvips + Acrobat Distiller + Acrobat Reader. Using other ways of PDF creation, like pdfLaTeX, are possible, but not required.

Disclaimer

This is a pre-alpha release of the `texpower` package, mainly for getting feedback on the `\stepwise` command.

During the subsequent error correction and extension of the functionality, the syntax and implementation of the macros described here are liable to change.

So far, the `texpower` package itself contains only scarce inline documentation, as the code is too much of a moving target to make rigorous documentation a sensible endeavour. As soon as the `texpower` package is ready for beta release, it will be made into a fully documented `dtx` file.

It is planned to add code for structured backgrounds until alpha release.

Credits

I am indepted to Klaus Guntermann for providing the motivation and the basis for this package. His Pdf Presentation Post Processor PPower4 (which I can't use, because PSTricks doesn't work with pdfLaTeX) prompted me to develop this package as an alternative, and his package `texpause` is the basis for the code of `texpower`.

Further thanks go to Marc van Dongen for allowing me to include his code for page transitions.

Contents

1 Examples

First, two simple examples for the `\pause` command.

All other examples are meant to illustrate the expressive power of the `\stepwise` command.

Looking at the code for the examples will probably be the best way of understanding how certain effects can be achieved.

1.1 Some examples for \pause

a

1.1 Some examples for \pause

a

b

1.1 Some examples for \pause

a

b

c

1.1 Some examples for \pause

a

b

c

- foo

1.1 Some examples for \pause

a

b

c

- foo
- bar

1.1 Some examples for \pause

a

b

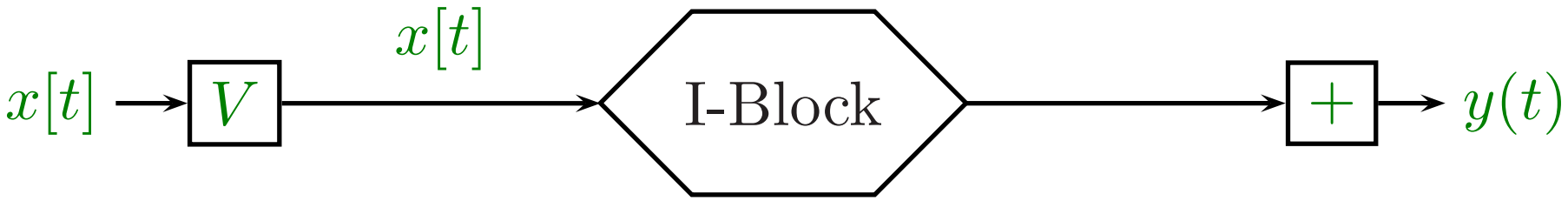
c

- foo
- bar
- baz

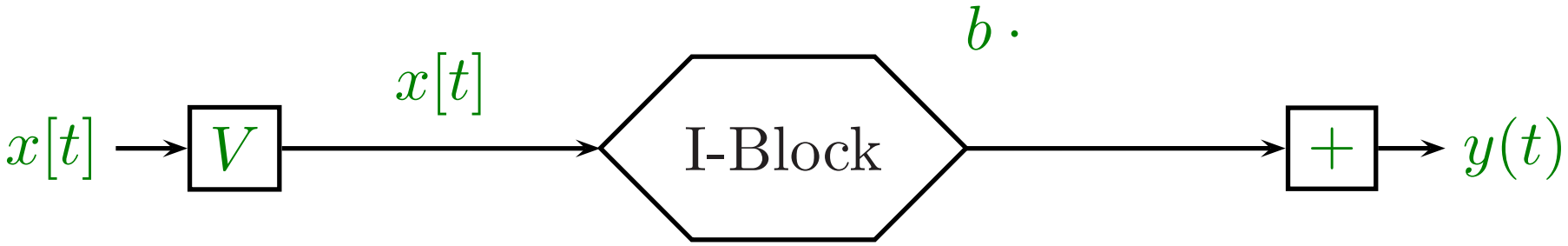
1.2 \stepwise Example: A Picture

$x[t] \longrightarrow$ $\longrightarrow y(t)$

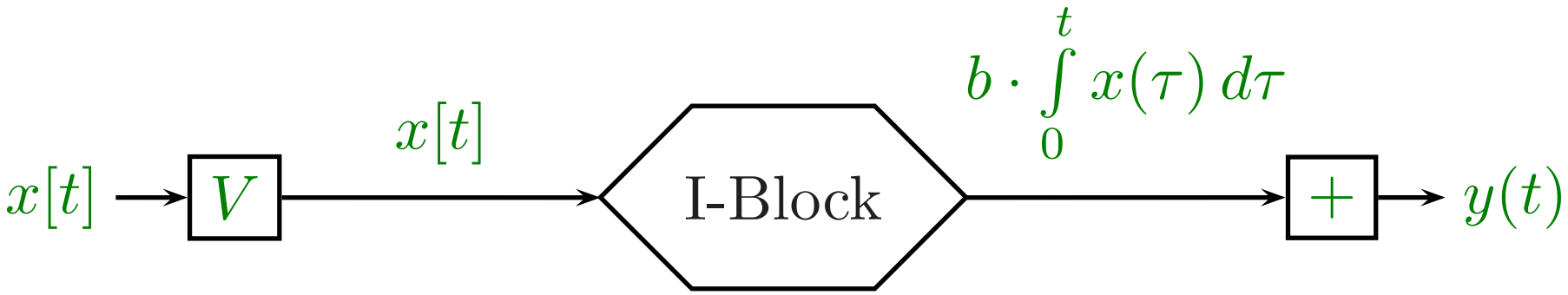
1.2 \stepwise Example: A Picture



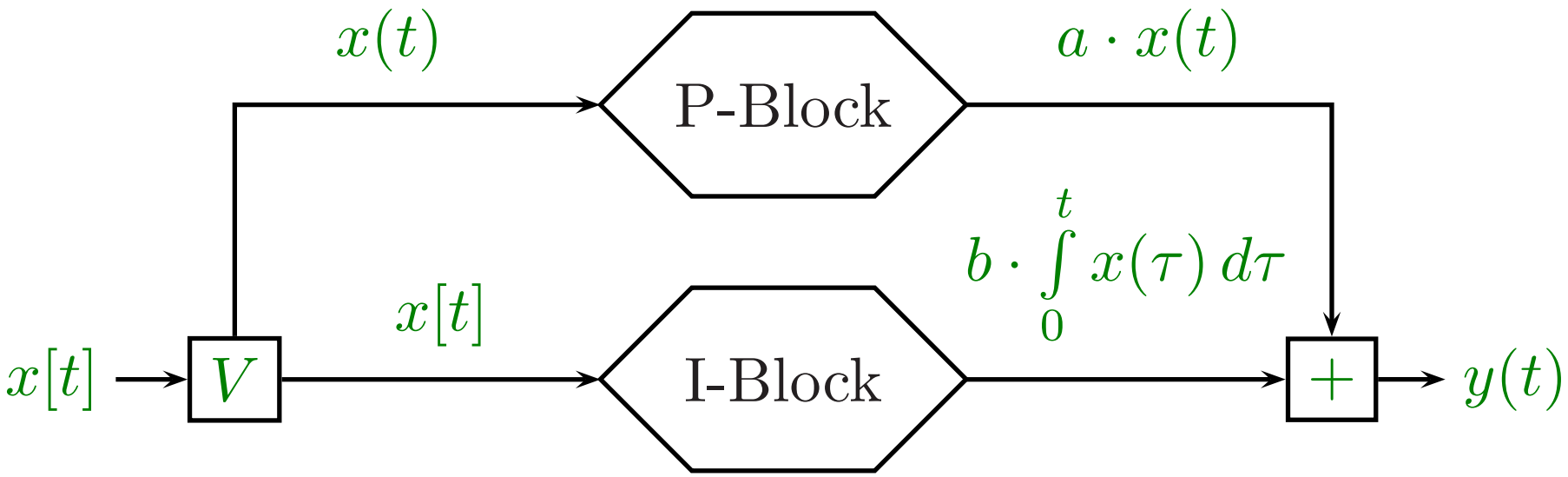
1.2 \stepwise Example: A Picture



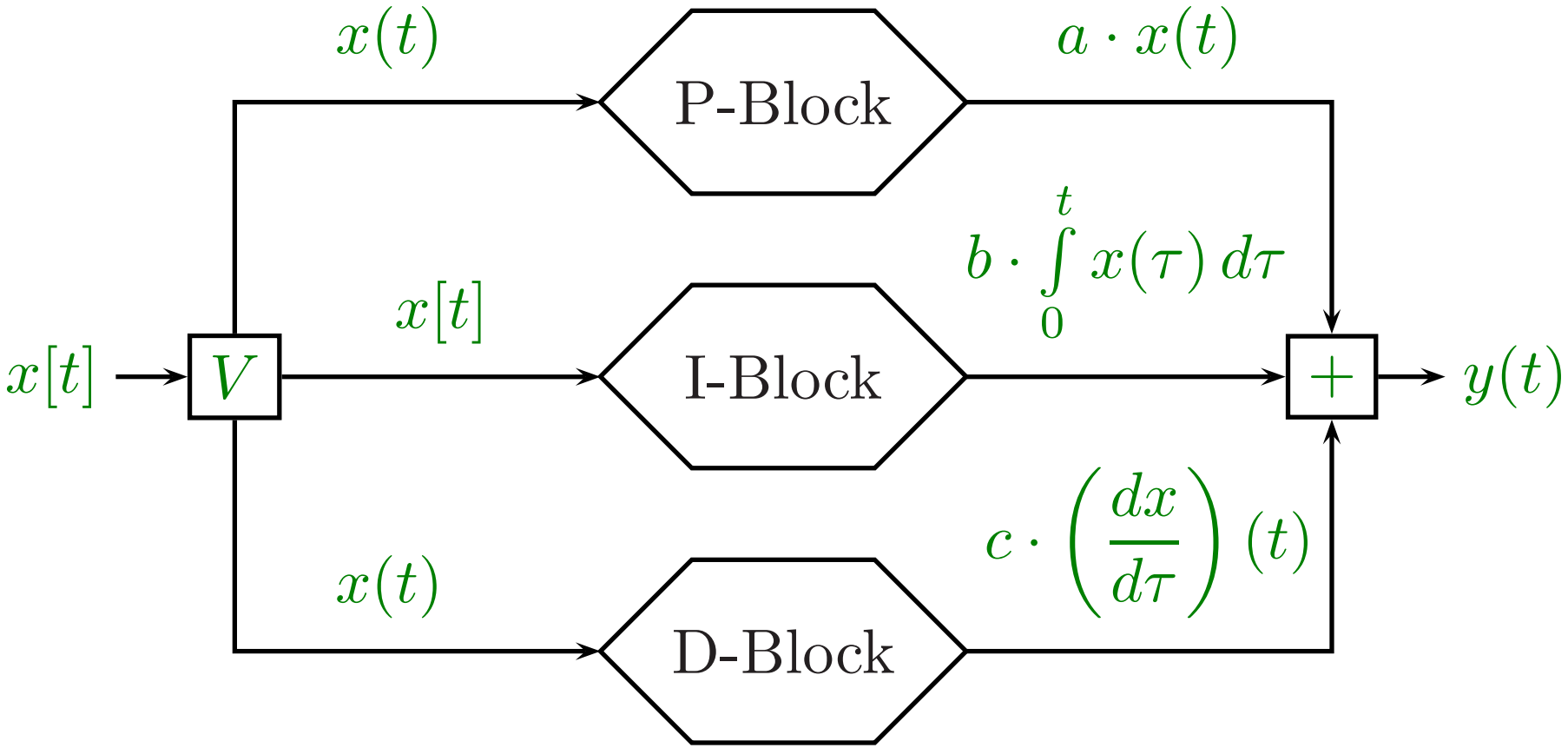
1.2 \stepwise Example: A Picture



1.2 \stepwise Example: A Picture



1.2 \stepwise Example: A Picture



1.3 \stepwise Example: A Tabular

They can	be built	line by line

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell		

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell
or		

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell
or	like	

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell
or	like	this.

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell
or	like	this.
But		

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell
or	like	this.
But	beware	

1.3 \stepwise Example: A Tabular

They can	be built	line by line
or cell	by	cell
or	like	this.
But	beware	of cells growing horizontally!

1.4 \stepwise Example: An Aligned Equation

$$\min \left(\qquad , \qquad \right)$$

1.4 \stepwise Example: An Aligned Equation

$$\min\left(\max\left(\begin{array}{l}\vdots\\ \end{array},\right),\right)$$

1.4 \stepwise Example: An Aligned Equation

$$\min\left(\begin{array}{l}\max\left(\begin{array}{l}\min\Big(F'(x),\min\big(F_1(x),G_1(y)\big)\Big),\\ \vdots\end{array}\right),\\ \end{array}\right.$$

1.4 \stepwise Example: An Aligned Equation

$$\min\left(\begin{array}{l}\max\left(\begin{array}{l}\min\Big(F'(x),\min\big(F_1(x),G_1(y)\big)\Big),\\ \vdots\\ \min\Big(F'(x),\min\big(F_n(x),G_n(y)\big)\Big)\end{array}\right),\\ \end{array}\right)$$

1.4 \stepwise Example: An Aligned Equation

$$\min\left(\begin{array}{l}\max\left(\begin{array}{l}\min\Big(F'(x),\min\big(F_1(x),G_1(y)\big)\Big),\\ \vdots\\ \min\Big(F'(x),\min\big(F_n(x),G_n(y)\big)\Big)\end{array}\right),\\ \min\big(G_i(y),H_i(z)\big)\end{array}\right)$$

1.4 \stepwise Example: An Aligned Equation

$$\min\left(\max\left(\begin{matrix}\min\left(F'(x),\min\left(F_1(x),G_1(y)\right)\right),\\ \vdots\\ \min\left(F'(x),\min\left(F_n(x),G_n(y)\right)\right)\end{matrix}\right),\min\left(G_i(y),H_i(z)\right)\right)$$

$$= \max\left(\begin{matrix}\min\left(\min\left(\quad,\min\left(\quad\right)\right),\min\left(G_i(y),H_i(z)\right)\right),\\ \vdots\\ \min\left(\min\left(\quad,\min\left(\quad\right)\right),\min\left(G_i(y),H_i(z)\right)\right)\end{matrix}\right)$$

$$= \max\left(\begin{matrix}\min\left(\min\left(\quad,\min\left(\quad,\min\left(\quad,G_i(y)\right)\right)\right),H_i(z)\right),\\ \vdots\\ \min\left(\min\left(\quad,\min\left(\quad,\min\left(\quad,G_i(y)\right)\right)\right),H_i(z)\right)\end{matrix}\right)$$

1.4 \stepwise Example: An Aligned Equation

$$\min\left(\begin{array}{l}\max\left(\begin{array}{l}\min\left(F'(x),\min\left(F_1(x),G_1(y)\right)\right),\\ \vdots\\ \min\left(F'(x),\min\left(F_n(x),G_n(y)\right)\right)\end{array}\right),\min\left(G_i(y),H_i(z)\right)\end{array}\right)$$

$$=\max\left(\begin{array}{l}\min\left(\min\left(F'(x),\min\left(F_1(x),G_1(y)\right)\right),\min\left(G_i(y),H_i(z)\right)\right),\\ \vdots\\ \min\left(\min\left(F'(x),\min\left(F_n(x),G_n(y)\right)\right),\min\left(G_i(y),H_i(z)\right)\right)\end{array}\right)$$

$$=\max\left(\begin{array}{l}\min\left(\min\left(F'(x),\min\left(F_1(x),\min\left(G_1(y),G_i(y)\right)\right)\right),H_i(z)\right),\\ \vdots\\ \min\left(\min\left(F'(x),\min\left(F_n(x),\min\left(G_n(y),G_i(y)\right)\right)\right),H_i(z)\right)\end{array}\right)$$

1.4 \stepwise Example: An Aligned Equation

$$
\begin{aligned}
&\min\left(\max\left(\begin{array}{l}\min\left(F'(x),\min\left(F_1(x),G_1(y)\right)\right),\\ \vdots\\ \min\left(F'(x),\min\left(F_n(x),G_n(y)\right)\right)\end{array}\right),\min\left(G_i(y),H_i(z)\right)\right)\\
&\quad=\max\left(\begin{array}{l}\min\left(\min\left(F'(x),\min\left(F_1(x),G_1(y)\right)\right),\min\left(G_i(y),H_i(z)\right)\right),\\ \vdots\\ \min\left(\min\left(F'(x),\min\left(F_n(x),G_n(y)\right)\right),\min\left(G_i(y),H_i(z)\right)\right)\end{array}\right)\\
&\quad=\max\left(\begin{array}{l}\min\left(\min\left(F'(x),\min\left(F_1(x),\min\left(G_1(y),G_i(y)\right)\right)\right),H_i(z)\right),\\ \vdots\\ \min\left(\min\left(F'(x),\min\left(F_n(x),\min\left(G_n(y),G_i(y)\right)\right)\right),H_i(z)\right)\end{array}\right)\\
&\quad=\min\left(F'(x),\min\left(\max\left(\begin{array}{l}\min\left(F_1(x),\min\left(G_1(y),G_i(y)\right)\right),\\ \vdots\\ \min\left(F_n(x),\min\left(G_n(y),G_i(y)\right)\right)\end{array}\right),H_i(z)\right)\right)
\end{aligned}
$$

1.5 \stepwise Example: Inside A Paragraph

We can _____ a

_____ _____

which is then ____ in in

____ order!

1.5 \stepwise Example: Inside A Paragraph

We can create a

_____ _____

which is then ____ in in

____ order!

1.5 \stepwise Example: Inside A Paragraph

We can create a fill-in-the-blanks _____ which is then _____ in in _____ order!

1.5 \stepwise Example: Inside A Paragraph

We can create a fill-in-the-blanks ____ which is then ____ in in **any** order!

1.5 \stepwise Example: Inside A Paragraph

We can create a fill-in-the-blanks ____ which is then filled in in **any** order!

1.5 \stepwise Example: Inside A Paragraph

We can create a fill-in-the-blanks text which is then filled in in **any** order!

1.6 \stepwise Example: Writing Backwards

1.6 \stepwise Example: Writing Backwards

possible!

1.6 \stepwise Example: Writing Backwards

to possible!


1.6 \stepwise Example: Writing Backwards

write

to possible!



1.6 \stepwise Example: Writing Backwards

backwards write

to possible!

1.6 \stepwise Example: Writing Backwards

it backwards write

to possible!





1.6 \stepwise Example: Writing Backwards

now it backwards write

to possible!






1.6 \stepwise Example: Writing Backwards

Is now it backwards write to possible!

1.7 \stepwise Example: Highlighting Text

Instead of displaying incrementally, we can just ‘flip through’ some items by highlighting them:

- Item 1
- Item 2
- Item 3

1.7 \stepwise Example: Highlighting Text

Instead of displaying incrementally, we can just ‘flip through’ some items by highlighting them:

- Item 1
- Item 2
- Item 3

1.7 \stepwise Example: Highlighting Text

Instead of displaying incrementally, we can just 'flip through' some items by highlighting them:

- Item 1
- Item 2
- Item 3

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 **2**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 **3**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 **4**

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 **5**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 3 **6**

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 **7**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 4 **8**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 3 **9**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 5 **10**

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 **11**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 3 4 6 **12**

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 **13**

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 7

14

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 3 5

15

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 4 8

16

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

17

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 3 6 9

18

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

19

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 4 5 10

20

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 3 7

21

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 11

22

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

23

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 3 4 6 8 12

24

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 5

25

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 13

26

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 3 9

27

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 4 7

14

28

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

29

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 3 5 6 10

15

30

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

31

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 4 8

16

32

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 3 11

33

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2

17

34

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 5 7

35

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 3 4 6 9 12

18

36

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1

37

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2

19

38

1.8 \stepwise Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 3 13

39

1.8 `\stepwise` Example: Fooling Around

'Tweaking' the hooks a little allows some truly bizarre applications...

Divisibility demo:

1 2 4 5 8 10

20

40

2 Documentation

For this pre-alpha release, the documentation section contains only rudimentary explanations of the basic functionality. So far, not even all the bells and whistles used for implementing the Examples section are explained. Please look at the comments inside this `.tex` file and inside the file `texpower.sty` to find out what's going on.

For the first alpha release, this documentation section will be completed. For the first beta release, when the code is a little more stable, the `texpower` package will be made into a properly documented `.dtx` file.

2.1 The \pause command

\pause works exactly as the \pause command from the Package texpause which is part of the PPower4 suite by Klaus Guntermann (in fact, the code is almost identical; thanks to Klaus for letting me use it).

In particular, it will end the current paragraph, ship out the current page, start a new page and copy whatever was on the current page onto the new page, where typesetting is resumed.

2.1 The \pause command

\pause works exactly as the \pause command from the Package texpause which is part of the PPower4 suite by Klaus Guntermann (in fact, the code is almost identical; thanks to Klaus for letting me use it).

In particular, it will end the current paragraph, ship out the current page, start a new page and copy whatever was on the current page onto the new page, where typesetting is resumed.

This will create the effect of a pause in the presentation, i. e. the presentation stops because the current page ends at the point where the \pause command occurred and is resumed at this point when the presenter switches to the next page.

Things to pay attention to

1. `\pause` should appear in vertical mode only, i. e. between paragraphs or at places where ending the current paragraph doesn't hurt.
2. This means `\pause` is forbidden in all boxed material (including `tabular`), headers/footers, and floats.
3. `\pause` shouldn't appear either in environments which have to be closed to work properly, like `picture`, `tabbing`, and (unfortunately) environments for aligned math formulas.
4. `\pause` does work in all environments which mainly influence paragraph formatting, like `center`, `quote` or all list environments.

5. `\pause` doesn't really have problems with automatic page breaking, but beware of overfull pages/slides. In this case, it may occur that only the last page(s)/slide(s) of a sequence are overfull, which changes vertical spacing, making lines 'wobble' when switching to the last page/slide of a sequence.

A lot of the restrictions for the use of pause can be avoided by using `\stepwise` (see next section).

2.2 The \stepwise command

\stepwise{⟨contents⟩} is a command for displaying some part of a LaTeX document (which is contained in ⟨contents⟩) 'step by step'. As of itself, \stepwise doesn't do very much. If ⟨contents⟩ contains one or more constructs of the form \step{⟨stepcontents⟩}, the following happens:

1. The current paragraph is ended.
2. The current contents of the page are saved (as with \pause).

3. As many pages as there are \step commands in ⟨**contents**⟩ are produced.

   Every page starts with what was on the current page when \stepwise started.

   The first page also contains everything in ⟨**contents**⟩ which is not in ⟨**stepcontents**⟩ for any \step command.

   The second page additionally contains the ⟨**stepcontents**⟩ for the first \step command, and so on, until all ⟨**stepcontents**⟩ are displayed.

4. When all ⟨**stepcontents**⟩ are displayed, \stepwise ends and typesetting is resumed (still on the current page).

This will create the effect that the \step commands are executed ‘ ’.

3. As many pages as there are \step commands in ⟨**contents**⟩ are produced.

   Every page starts with what was on the current page when \stepwise started.

   The first page also contains everything in ⟨**contents**⟩ which is not in ⟨**stepcontents**⟩ for any \step command.

   The second page additionally contains the ⟨**stepcontents**⟩ for the first \step command, and so on, until all ⟨**stepcontents**⟩ are displayed.

4. When all ⟨**stepcontents**⟩ are displayed, \stepwise ends and typesetting is resumed (still on the current page).

This will create the effect that the \step commands are executed ‘step ’.

3. As many pages as there are \step commands in ⟨**contents**⟩ are produced.

   Every page starts with what was on the current page when \stepwise started.

   The first page also contains everything in ⟨**contents**⟩ which is not in ⟨**stepcontents**⟩ for any \step command.

   The second page additionally contains the ⟨**stepcontents**⟩ for the first \step command, and so on, until all ⟨**stepcontents**⟩ are displayed.

4. When all ⟨**stepcontents**⟩ are displayed, \stepwise ends and typesetting is resumed (still on the current page).

This will create the effect that the \step commands are executed ‘step by ’.

3. As many pages as there are \step commands in ⟨**contents**⟩ are produced.

   Every page starts with what was on the current page when \stepwise started.

   The first page also contains everything in ⟨**contents**⟩ which is not in ⟨**stepcontents**⟩ for any \step command.

   The second page additionally contains the ⟨**stepcontents**⟩ for the first \step command, and so on, until all ⟨**stepcontents**⟩ are displayed.

4. When all ⟨**stepcontents**⟩ are displayed, \stepwise ends and typesetting is resumed (still on the current page).

This will create the effect that the \step commands are executed 'step by step'.

Things to pay attention to

1. `\stepwise` should appear in vertical mode only, i. e. between paragraphs, just like `\pause`.

2. Don't put `\pause` or nested occurrences of `\stepwise` into ⟨**contents**⟩.

3. Structures where `\pause` does not work (like `tabular` or aligned equations) can go completely into ⟨**contents**⟩, where `\step` can be used freely (see Examples section).

4. As ⟨**contents**⟩ is read as a macro argument, constructs involving catcode changes (like `\verb` or language switches) won't work in ⟨**contents**⟩. Owing to a great suggestion by Ross Moore, I hope to remedy this at least to some extent until the alpha release.

5. Several instances of \stepwise may occur on one page, also combined with \pause (outside of ⟨**contents**⟩).

   But beware of page breaks in ⟨**contents**⟩. This will really mess things up. Overfull pages/slides are also a problem, just like with \pause.

6. \step can go in ⟨**stepcontents**⟩. The order of execution of \step commands is just the order in which they appear in ⟨**contents**⟩, independent of nesting within each other.

7. Beware of things using global counters, accessing files etc. in ⟨**contents**⟩, because everything is executed several times. This means sections, equation numbers, labels, hyperlinks and the like.

   While labels and hyperlinks work sort of (giving a lot of warnings though), global counters just go astray.

`\stepwise` is meant for complex structured objects like pictures, equations or tabulars, where `\pause` can't go. I don't know whether this problem is worth going into a lot of trouble.

Page numbers are taken care of explicitly, as with `\pause`. This would probably work for other counters, too. Also, `amsmath` seems to save and restore all counters for its measuring pass, so maybe this could be done for `\stepwise` as well.

I'll try to get equation numbers right for the alpha release, but go out of my way further only if I see what it's good for.

2.2.1 \boxedsteps and \nonboxedsteps

By default, ⟨**stepcontents**⟩ belonging to a \step which is not yet 'active' are ignored altogether. This makes it possible to include e. g. tabulators & or line breaks into ⟨**stepcontents**⟩ without breaking anything.

Sometimes, however, this behaviour is undesirable, for instance when stepping through an equation 'from outer to inner', or when filling in blanks in a paragraph. Then, the desired behaviour of a \step which is not yet 'active' is to create an appropriate amount of blank space where ⟨**stepcontents**⟩ can go as soon as it is activated.

This behaviour is toggled by the following commands:

`\boxedsteps` makes `\step` leave blank space the size of ⟨**stepcontents**⟩ when inactive and put ⟨**stepcontents**⟩ into a box when active.

`\nonboxedsteps` makes `\step` ignore ⟨**stepcontents**⟩ when inactive and leave ⟨**stepcontents**⟩ alone when active (default).

Things to pay attention to

1. The settings effected by `\boxedsteps` and `\nonboxedsteps` are local, i. e. whenever a group closes, the setting is restored to its previous value.

2. Putting stuff into boxes can break things like tabulators (`&`). It can also mess up math spacing, which then has to be corrected manually. Compare the following examples:

$$\left(\frac{a+b}{c}\right) \qquad \left(\frac{a}{c}\right) \qquad \left(\frac{a}{c}\right)$$

Things to pay attention to

1. The settings effected by `\boxedsteps` and `\nonboxedsteps` are local, i. e. whenever a group closes, the setting is restored to its previous value.

2. Putting stuff into boxes can break things like tabulators (`&`). It can also mess up math spacing, which then has to be corrected manually. Compare the following examples:

$$\left(\frac{a+b}{c}\right) \qquad \left(\frac{a+b}{c}\right) \qquad \left(\frac{a+b}{c}\right)$$

2.2.2 Custom versions of \stepwise

If ⟨**contents**⟩ contains several paragraphs, it might happen that vertical spacing is different on every page of a sequence generated by \stepwise,[a] making lines 'wobble'.

There are two custom versions of \stepwise which produce proper vertical spacing by enclosing ⟨**contents**⟩ in a minipage.

\liststepwise{⟨**contents**⟩} works exactly like \stepwise, but ⟨**contents**⟩ is enclosed in a minipage. Use for list environments and aligned equations.

\parstepwise{⟨**contents**⟩} works like \liststepwise, but \boxedsteps is turned on by default. Use for texts where \steps are to be filled into blank spaces.

[a] I absolutely don't know why this is so, as every page of the sequence is delimited by \vfill. Maybe this can be remedied.

2.2.3 Starred versions of `\stepwise` commands

Usually, the first page of a sequence produced contains only material which is not part of any ⟨**stepcontents**⟩. The first ⟨**stepcontents**⟩ are displayed on the second page of the sequence.

For special effects (see example 1.7), it might be desirable to have the first ⟨**stepcontents**⟩ active even on the first page of the sequence.

All variants of `\stepwise` have a starred version (e. g. `\stepwise*`) which does exactly that.

2.2.4 The optional argument of \stepwise

Every variant of \stepwise takes an optional argument, like this

\stepwise[⟨settings⟩]{⟨contents⟩}.

⟨settings⟩ will be placed right before the internal loop which produces the sequence of pages. It can contain settings of parameters which modify the behaviour of \stepwise or \step. ⟨settings⟩ is placed inside a group so all changes are local to this call of \stepwise.

Some internal macros and counters which can be adjusted are explained in the following.

2.2.5 Customizing the way ⟨stepcontents⟩ is diplayed

Internally, there are three macros (taking one argument each) which control how ⟨**stepcontents**⟩ is displayed: `\displaystepcontents`, `\hidestepcontents`, and `\activatestep`. Virtually, every `\step{`⟨**stepcontents**⟩`}` is replaced by

`\hidestepcontents{`⟨**stepcontents**⟩`}` when this step is not yet active.

`\displaystepcontents{\activatestep{`⟨**stepcontents**⟩`}}` when this step is activated for the first time.

`\displaystepcontents{`⟨**stepcontents**⟩`}` when this step has been activated before.

By redefining these macros, the behaviour of \step is changed accordingly. You can redefine them inside ⟨**contents**⟩ to provide a change affecting one \step only, or in the optional argument of \stepwise to provide a change for all \steps inside ⟨**contents**⟩.

texpower offers four standard definitions:

\displayidentical Simply expands to its argument. The same as LaTeXs \@ident. Used by \nonboxedsteps (default).

\displayboxed Expands to an \mbox containing its argument. Used by \boxedsteps.

\hideignore Expands to nothing. The same as LaTeXs \@gobble. Used by \nonboxedsteps (default).

\hidephantom Expands to a \phantom containing its argument. Used by \boxedsteps.

\activatestep is set to \displayidentical by default.

In the Examples section, it is demonstrated how special effects can be achieved by redefining these macros.

2.2.6 \restep

Frequently, it is desirable for two or more steps to appear at the same time, for instance to fill in arguments at several places in a formula at once (see example 1.4).

`\restep{⟨stepcontents⟩}` is identical to `\step{⟨stepcontents⟩}`, but is activated at the same time as the previous occurrence of `\step`.

2.2.7 Optional arguments of \step

Sometimes, letting two \steps appear at the same time (with \restep) is not the only desirable modification of the order in which \steps appear. \step takes two optional arguments for influencing the mode of activation, like this:

\step[⟨**activatefirst**⟩][⟨**whenactive**⟩]{⟨**stepcontents**⟩}.

Both ⟨**activatefirst**⟩ and ⟨**whenactive**⟩ should be conditions in the syntax of the \ifthenelse command (see the documentation of the ifthen package for details).

⟨**activatefirst**⟩ checks whether this \step is to be activated for the first time. The default value is \value{step}=\value{stepcommand} (see section 2.2.8 for a list of internal values). By using \value{step}=⟨*n*⟩, this \step can be forced to appear as the *n*th one. See example 1.5 for a demonstration of how this can be used to make \steps appear in arbitrary order.

⟨**whenactive**⟩ checks whether this \step is to be considered active at all. The default behaviour is to check whether this \step has been activated before (this is saved internally for every step). See example 1.8 for a demonstration of how this can be used to make \steps appear and disappear after a defined fashion.

If you know what you're doing...

Both optional arguments allow two syntctical forms:

1. enclosed in square brackets `[]` like explained above.

2. enclosed in braces `()`. In this case, ⟨**activatefirst**⟩ and ⟨**whenactive**⟩ are not treated as conditions in the sense of `\ifthenelse`, but as conditionals like those used internally by LaTeX. That means, ⟨**activatefirst**⟩ (when enclosed in braces) can contain arbitrary TeX code which then takes two arguments and expands to one of them, depending on whether the condition is fulfilled or not fulfilled. For instance, `\step[`⟨**activatefirst**⟩`]{`⟨**stepcontents**⟩`}` could be replaced by `\step(\ifthenelse{`⟨**activatefirst**⟩`}){`⟨**stepcontents**⟩`}`. See example 1.6 for a simple application of this syntax.

Internally, the default for the treatment of ⟨**whenactive**⟩ is `(\if@first@TP@true)`, where `\if@first@TP@true` is an internal condition checking whether this `\step` has been activated before.

2.2.8 Finding out what's going on

Inside ⟨**settings**⟩ and ⟨**contents**⟩, you can refer to the following internal state variables which provide information about the current state of the process executed by `\stepwise`:

counter: `firststep` The number from which to start counting steps (see counter `step` below). Is 0 by default and 1 for starred versions (section 2.2.3) of `\stepwise`. You can set this in ⟨**settings**⟩ for special effects (see example 1.6).

counter: `totalsteps` The total number of `\step` commands occurring in ⟨**contents**⟩.

counter: `step` The number of the current iteration, i. e. the number of the current page in the sequence of pages produced by `\stepwise`. Runs from `\value{firststep}` to `\value{totalsteps}`.

counter: `stepcommand` The number of the `\step` command currently being executed.

boolean: `firstactivation` `true` if this `\step` is active for the first time, `false` otherwise.

boolean: `active` `true` if this `\step` is currently active, `false` otherwise.

`stepcommand`, `firstactivation`, and `active` are useful only inside ⟨**stepcontents**⟩.

2.2.9 \afterstep

It might be neccessary to set some parameters which affect the appearance of the page (like page transitions) inside ⟨**stepcontents**⟩. However, the \step commands are usually placed deeply inside some structure, so that all local settings are likely to be undone by groups closing before the page is completed.

\afterstep{⟨**settings**⟩} puts ⟨**settings**⟩ right before the end of the page, after the current step is performed.

Things to pay attention to

1. There can be only one effective value for ⟨**settings**⟩. Every occurrence of \afterstep overwrites this value globally.

2. \afterstep will not be executed in ⟨**stepcontents**⟩ if the corresponding \step is not active, even if ⟨**stepcontents**⟩ is displayed owing to a redefinition of \nodisplay, like in example 1.7.

3. As ⟨**settings**⟩ is put immediately before the page break, there is no means of restoring the original value of whatever has been set from inside ⟨**contents**⟩. So if you set some page transition via \afterstep and want it to be reset in some later step, you have to reset it explicitly with another call of \afterstep.

2.3 Page Transitions

I am indepted to Marc van Dongen for allowing me to include a suite of commands written by him and posted to the PPower4 mailing list which set page transitions (using `hyperrefs \hypersetup`).

The following page transition commands are defined:

2.3 Page Transitions

I am indepted to Marc van Dongen for allowing me to include a suite of commands written by him and posted to the PPower4 mailing list which set page transitions (using hyperrefs \hypersetup).

The following page transition commands are defined:

\pageTransitionSplitHO	Split Horizontally to the outside.

2.3 Page Transitions

I am indepted to Marc van Dongen for allowing me to include a suite of commands written by him and posted to the PPower4 mailing list which set page transitions (using hyperrefs \hypersetup).

The following page transition commands are defined:

`\pageTransitionSplitHO`	Split Horizontally to the outside.
`\pageTransitionSplitHI`	Split Horizontally to the inside.

2.3 Page Transitions

I am indepted to Marc van Dongen for allowing me to include a suite of commands written by him and posted to the PPower4 mailing list which set page transitions (using hyperrefs \hypersetup).

The following page transition commands are defined:

\pageTransitionSplitHO	Split Horizontally to the outside.
\pageTransitionSplitHI	Split Horizontally to the inside.
\pageTransitionSplitVO	Split Vertically to the outside.

2.3 Page Transitions

I am indepted to Marc van Dongen for allowing me to include a suite of commands written by him and posted to the PPower4 mailing list which set page transitions (using hyperrefs \hypersetup).

The following page transition commands are defined:

\pageTransitionSplitHO	Split Horizontally to the outside.
\pageTransitionSplitHI	Split Horizontally to the inside.
\pageTransitionSplitVO	Split Vertically to the outside.
\pageTransitionSplitVI	Split Vertically to the inside.

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBoxO` Box growing to the outside.

`\pageTransitionBlindsH`	Horizontal Blinds.
`\pageTransitionBlindsV`	Vertical Blinds.
`\pageTransitionBoxO`	Box growing to the outside.
`\pageTransitionBoxI`	Box growing to the inside.

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBoxO` Box growing to the outside.

`\pageTransitionBoxI` Box growing to the inside.

`\pageTransitionWipe{`⟨**angle**⟩`}`

Wipe from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to wipe.

Apparently, only the values 0,

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBoxO` Box growing to the outside.

`\pageTransitionBoxI` Box growing to the inside.

`\pageTransitionWipe{`⟨**angle**⟩`}`

Wipe from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to wipe.

Apparently, only the values 0, 90,

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBoxO` Box growing to the outside.

`\pageTransitionBoxI` Box growing to the inside.

`\pageTransitionWipe{`⟨**angle**⟩`}`

Wipe from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to wipe.

Apparently, only the values 0, 90, 180,

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBoxO` Box growing to the outside.

`\pageTransitionBoxI` Box growing to the inside.

`\pageTransitionWipe{`⟨**angle**⟩`}`

Wipe from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to wipe.

Apparently, only the values 0, 90, 180, 270 are supported.

`\pageTransitionBlindsH` Horizontal Blinds.

`\pageTransitionBlindsV` Vertical Blinds.

`\pageTransitionBoxO` Box growing to the outside.

`\pageTransitionBoxI` Box growing to the inside.

`\pageTransitionWipe{`⟨**angle**⟩`}`

Wipe from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to wipe.

Apparently, only the values 0, 90, 180, 270 are supported.

`\pageTransitionDissolve` Dissolve.

`\pageTransitionGlitter{⟨angle⟩}`

Glitter from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to glitter.

Apparently, only the values 0,

`\pageTransitionGlitter{⟨angle⟩}`

Glitter from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to glitter.

Apparently, only the values 0, 270,

`\pageTransitionGlitter{⟨angle⟩}`

Glitter from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to glitter.

Apparently, only the values 0, 270, 315 are supported.

`\pageTransitionGlitter{⟨angle⟩}`

Glitter from one edge of the page to the facing edge.

⟨**angle**⟩ is a number between 0 and 360 which specifies the direction (in degrees) in which to glitter.

Apparently, only the values 0, 270, 315 are supported.

`\pageTransitionReplace` Simple Replace (the default).

Things to pay attention to

1. The setting of the page transition is a property of the page, i. e. whatever page transition is in effect when a page break occurs, will be assigned to the corresponding pdf page.

2. The setting of the page transition achieved by the `\pageTransition...` commands (via `\hypersetup`) is local, i. e. whenever a group ends, it will be restored to its original value before the beginning of the group.

3. Setting page transitions works well with `\pause`. Here, `\pause` acts as a page break, i. e. a different page transition can be set before every occurrence of `\pause`.

   Page transitions can also be set in the argument of `\step`, with the effect that the page transition becomes effective as soon as the respective `\step` is first displayed. However,

\steps frequently occur inside some kind of structure, so it is quite likely that some group will be closed before the page break occurs, undoing the setting. Hence, it is recommended to use i. e. \afterstep{\pageTransitionDissolve} to set the page transition, which will put the page transition setting right before the page break (see example 1.2).