Custom-numbered-blocks Example

1 Custom blocks and crossreferencing

With this filter, you can define custom div classes (environments) that come with numbering, such as theorems, examples, exercises. The filter supports output formats pdf and html.

Feature 1.1: Numbering: new, adapted to quarto 1.4 cross reference overhaul

Numbering is by default within document and without section prefix for single documents, or within chapter and with chapter prefix for books. Grouped classes share the same counter, and the same default style.
The default behaviour can be changed by setting the item `chapters` in yaml-lkey `crossreference` to `true` or `false`, to comply with quarto's (no longer so) new crossreferencing options.
In the present single document, yaml contains

```
crossref:
  chapters: true
```

Numbered custom blocks can be cross-referenced with `\ref`.
Default numbering can be switched off for the whole class by setting the `numbered` to false, or for an individual block by adding the class `unnumbered`.
Crossreferences my need a re-run to update.

Feature: Boxes can be nested

However, inner boxes are not numbered – it would be hard to put them in a sequence with outer boxes anyway.

Feature 1.2: Block box style

The default style for custom divs is `foldbox`: a collapsible similar to quarto's callouts, with a collapse button at the bottom that makes it easier collapse long boxes, and box open to the right. It comes with the variant `foldbox.simple`, with closed box and no additional close button.

To do 1.1: Custom styles

- ☐ create an API for user defined block styles
- ☐ provide an example
- ☐ and documentation

Done, may change 1.2: Custom list of blocks

Generate `.qmd` files that contains a list of selected block classes, intermitted with headers from the document for easy customization and commenting. This way, one can make a list of all definitions, examples, or {theorems, propositions and lemmas} etc., edit it later and attach to the main document. If you edit, make sure to rename the autogenerated list first, otherwise it will get overwritten in the next run and all work is lost …
Currently, you need to give a key `listin` for any class or group of classes that should appear in a list of things. The value of this key is an array of names, also if only one list is to be generated. These names are turned into files `list-of-name.qmd`. I am considering replacing the yaml interface by a sub-key to `custom-numbered-classes`. This would allow to define arbitrary classes that can be attached to any custom div block, such as `.important`.

2 Pseudomath examples

Definition Pseudo.1: Fancybox

A box is called *fancybox* if it looks quite fancy.
In this context, by *fancy* we mean that the title of the box appears as a clickable button when rendered as html, where *clickable* implies that it throws a small shadow that becomes bigger when hovering over it.

Corollary Pseudo.2

By Definition Pseudo.1, `foldboxes` are fancyboxes.

Conjecture Pseudo.3

Students are lured into clicking on the title and unfolding the fancybox.

Theorem Pseudo.4

This extension has been written by a teacher who hopes that students read the course notes...

Theorem Pseudo.4 is suggested by Conjecture Pseudo.3, but it cannot be proven theoretically. It does give rise to more conjectures, though.

Conjecture

The teacher mentioned in Theorem Pseudo.4 is a statistician who got addicted to quarto due to James J Balamuta's fantastic web-r extension, and desparately wanted to have a common counter for theorems and alike. She got also convinced that everything is possible in quarto by the many nice extensions from Shafayet Khan Shafee.