1 Introduction

When writing a longer document, it is often desirable to compile only a part of the document, and there are several ways to achieve that. This example uses the `docmute` package to achieve that. But what we really want to know here is: Can we use pictures externalized when compiling the included files when we compile the main document, and vice versa? Sure!

The template for names of memo files is not fixed. It is composed of:

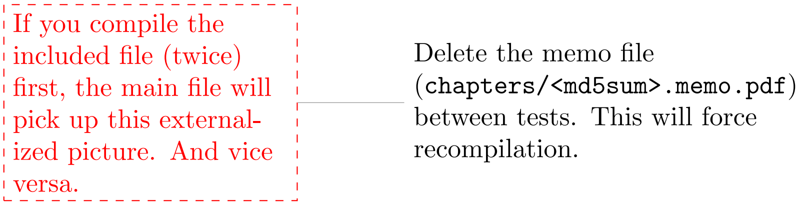
1. the prefix which you can set by memoize key `memo filename prefix`,
2. the automatically computed md5sum, and
3. the suffix which you can set by memoize key `memo filename suffix`.

Now crucially, the prefix can contain a path (which must exist, in the usual setup TeX can't create folders for you). In this document, we have exploited this to set

- `memo filename prefix={chapters/book.memo.dir/}` in the main file — note the braces around the value, and the slash at the end! — and
- `memo filename prefix={book.memo.dir/}` in the included file, which resides in directory `chapters`.

2 Included

The code for this section was input by `\input{chapters/4-chapter}`.



If you compile the included file (twice) first, the main file will pick up this externalized picture. And vice versa.

Delete the memo file (`chapters/<md5sum>.memo.pdf`) between tests. This will force recompilation.

3 Readonly

When `\memoize{readonly}` is in effect, memoize will use any pictures that were already externalized, but it will not externalize new stuff. This is great when we work on a picture, because we don't want to see it as an extra page all the time (and wait for the old version to be split off all the time).

One possible workflow is to say `readonly` in the included file, as that is the file that you will be compiling over and over again when you work on it. Nothing will get externalized, but the compilation will be fast, because externalized pictures will be used. Once you want to look at the whole book, you compile the main file (which contains no `readonly`) and everything gets externalized.

4 Conclusion

Easy, right?