Publishing language-specific PDFs

with Python code

Table of contents

As discussed on the previous section, the Quarto default is to render tabset languages as a single PDF. This is less than ideal. In this section I explain a work around. It involves adding a pre-render script that transforms tagged .qmd files into language specific files.

The script removes all tabsets from language specific files and renders them only as a PDF. The PDFs are then linked to the multilingual html.

Pre-render

I have attached the python script as a link. The file searches for any .qmd file ending with `-multi.qmd`. It then creates language-specific versions of each file stored in a folder named after that language.

The languages included are: R, Stata, Python, and Julia. If a language is not identified in the file, no language-specific file is created. Additional languages can easily be added by editing the pre-render script. Note, the term 'language' here really refers to the tabset labels.

The file should ignore tabsets that do not include labels amongst the specified languages. Alternatively, one could edit the script to search only tabsets with a specific tabset group (i.e., `group="languages"`). However, this will complicate instances where other tabsets include codeblocks that then do not match the engine specified in the single-language files.

The script needs to be run before rendering the project. Edit your _quarto.yml as follows:

```{r}
project:
  type: website
  output-dir: docs
  pre-render: _pre-render.py
```

Recall, we used the `engine: knitr` to render these multilingual files. In addition, we needed to load Statamarkdown to run Stata:

```{r multisetup}
#| include: false
library(Statamarkdown)
knitr::opts_chunk$set(collectcode = TRUE)
```

I have specifically named this codeblock `multisetup` (setup specific to the use of multiple languages) so that it can be easily identified and removed from language-specific files. The code also replaces the `engine: knitr` with language-specific engines:

- R: `engine: knitr`
- Stata: `jupyter: nbstata`
- Python: `jupyter: python3`
- Julia: `engine: julia`

The code also adds the subtitle "with [language] code" to the published PDF.

> ⚠ Existing files
>
> The pre-render script does not delete all .qmd files in the language-specific sub-directories at the start. For this reason, if you delete a -multi.qmd file from the main folder, you will also need to manually delete all language-specific versions.

Linking PDFs

You need to manually add links to the language-specific files in the -multi.qmd file. Note, these links are stripped from the language-specific .qmd files, which are only rendered as PDFs.

I was not able to get this functionality to work with Quarto's `format-links`. It's not clear why, as the documentation suggests that they can be quite flexible. In the end, I switched to adding `other-links` under the html format.

The YAML of your `-multi.qmd` files should look something like:

```yaml
#| eval: false
---
format
  html:
    toc: true
    other-links:
      - text: "PDF with R"
        href: "R/my-file-R.pdf"
        icon: file-pdf
      - text: "PDF with Stata"
        href: "Stata/my-file-Stata.pdf"
        icon: file-pdf
      - text: "PDF with Python"
        href: "Python/my-file-Python.pdf"
        icon: file-pdf
---
```

where `my-file` is a place holder for the file name ending in `-multi.qmd`.

> **File name changes**
>
> If you change the name of your main .qmd file, you also need to change the name of the corresponding links in the YAML.

Proof of concept

Below I run have two simple tabsets, each of which will appear in their language-specific folder. I have excluded any Julia code.

```python
print("Hello, Python world!")
```

```
Hello, Python world!
```

Second tabset:

```python
print("Hello again, Python world!")
```

```
Hello again, Python world!
```

Take a look at the attached files.