Unilur Example

Usage once extension installed

- **Activate** the extension by adding the following lines to your YAML header:

```
format:
  unilur-html: default
  unilur-html+solution:
    # You have to specify a different output file otherwise they will
    # overwrite themselves
    output-file: example-solution.html
```

When you call `quarto render`, two HTML files will be produced. The one named `example-solution.html` will have the solution blocks included.

- **Solution** code blocks are **highlighted** or ~~discarded~~ according to the chunk option `unilur-solution` Boolean and the global option `show-solution` in the YAML header.

Of note, if `show-solution` is absent, it is considered `false`.

Example with `unilur-solution: true`

- Code chunk with `echo: fenced`

> **Solution**
>
> ```{r}
> #| unilur-solution: true
>
> 1 + 1
> ```
>
> [1] 2

- Code block *without* solution, so stays in whatever happens

```{r}
1 + 1
```

`[1] 2`

- Solution with `unilur-collapse: false`

> **Solution**
>
> ```
> 1 + 2
> ```
>
> `[1] 3`

- Solution with markdown text (`{block}`):

> **Solution**
>
> For text with **bold** or warning people about CO_2 atmospheric concentration
>
> ```{r}
>
> 1 + 1
> ```

Any machinery supported by Quarto can be used:

- Python Code chunk with `unilur-solution: true`

> **Solution**
>
> ```
> for i in [3, 5, 6]:
>   print(i)
> ```
>
> ```
> 3
> 5
> 6
> ```

You can also include `div` labels with `unilur-solution`. Note the solution will only show up in the solution file.

> **Solution**
>
> Here is a solution that is inside a div. The contents of this will only be shown in the solution document. Source code highlighting is preserved inside the `div`.
>
> ```r
> df <- sample(mtcars,  size = 5)
> head(df)
> ```

The advantage of working inside the div labels is that standard code formatting and highlighting will apply, and code can be executed alongside standard text.