Demonstration Beamer/Reveal

Igor M. Coelho

April 29, 2020

Section 1

Part 4: Code

Code Embedding

Traditional code embedding can be done as in markdown or `beamer` listings. An interesting aspect is to have the code *actually executed*, and its output appended here.

One example of code in Python (not executed):

```
print("Hello World")
```

This is how it's done:

```
\```{.python}
print("Hello World")
\```
```

Configuring highlighters

Command `pandoc --list-highlight-styles` (or just `eval "$(pandoc --bash-completion)"` to enable bash completion) will display options: *pygments*, *tango*, *espresso*, *zenburn*, *kate*, *monochrome*, *breezedark*, *haddock*

You can visually compare them here: https://www.garrickadenbuie.com/blog/pandoc-syntax-highlighting-examples

An interesting project for active code execution is codebraid for: *Python 3.5+, Julia, Rust, R, Bash, JavaScript, and SageMath*.

Fenced code blocks

`fenced_code_blocks` and `backtick_code_blocks` enable code highlight on pandoc.

`fenced_code_attributes` can add more attributes to them.

See Manual:
https://pandoc.org/MANUAL.html#fenced-code-blocks

`pandoc --list-highlight-languages` will list languages (a lot!)

Active Code

To have code executed, you will need the plugin `pandoc-source-exec` from `panflute` pack (install as easy as `python3 -m pip install pandoc-source-exec`). On Pandoc, we can use `.exec` property (also `hide=true` and `.hide`). An specific command can be given, such as: `cmd='codes/run_python2.sh'`.

This approach is nice because it integrates nicely with `markdown-preview-enhanced` on Atom, where you can see results on real time.

Example of Active Code (Python3)

~~~~
\```{.python .exec cmd='codes/run_python3.sh'}
...
\```
~~~~

Expected output is:

```
Hello
World2
```

Real test

```python
print("Hello")
x=2
if x > 1:
  print("World"+str(x))
```

Output:

```
Hello
World2
```

Example of Active Code hidden (Python2)

```
\```{.python .exec cmd='codes/run_python2.sh'
                                   hide=true .hide output_label=''}
print 'Hello'
\```
```

Real test

```
Hello
```

Example of Active Code from file (Python3)

This is one of the most interesting approaches, just load script `codes/example.py` and execute it:

~~~~
\```{.python .exec cmd='codes/run_python3.sh' args="codes/example.py"
                                              hide=true .hide output_label=''}
\```
~~~~

File `example.py` contains:

```python
print("Example Hello World")
```

Real test

```
Example Hello World
```

Example of C++ Active Code

This is where we wanted to arrive on `codes/example.cpp`:

~~~~
\```{.cpp .exec cmd='codes/run_cpp.sh' args="codes/example.cpp"
                                  hide=true .hide output_label=''}
\```
~~~~

File `example.cpp` contains:

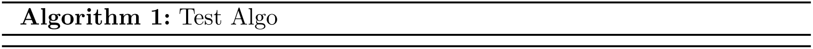
```cpp
#include<iostream>
int main() {
  std::cout << "Hello World C++" << std::endl;
  return 0;
}
```

Real test

```
Hello World C++
```

Pseudocode with LaTeX

LaTeX requires installation of `pdf2svg` for both `markdown-preview-enhanced` plugin and `run_latex.sh`.



Algorithm 1: Test Algo

Figure 1: Algo

~~~~
\```{.latex .exec .hide hide=true cmd='codes/run_latex.sh'
          void=true  args="figs/pseudo1.svg" output_label=''}
\documentclass{standalone}
\usepackage[ruled,vlined,linesnumbered]{algorithm2e}
\begin{document}
\pagestyle{empty}
\begin{algorithm}[H]
\caption{Test Algo}
\end{algorithm}
\end{document}
~~~~

Complete test of algorithm

```
Algorithm: While loop with If/Else condition
  Result: Write here the result
  Input   : Write here the input
  Output: Write here the output

  while While condition do
      instructions
      if condition then
          instructions1
          instructions2
      else
          instructions3
      end
  end
```

Figure 2: Algo

Ace Editor (on Atom)

For `markdown-preview-enhanced`, we can see Ace Editor (also on `revealjs`):

Section 2

Finish

Try more code types

Feel free to try other code types.

Ideally, we should be able to execute online code, but currently we lack a stable backend to integrate here (and a proper reward system to keep it up).

Learning more

Please contribute with us if you find more nice things!