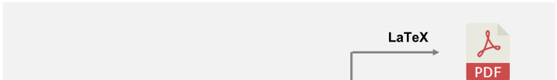
Christophe Dervieux
userR!2022 - 22nd of June

A tour of {knitr} engines: {knitr} not only knits R

{knitr} in the the R Markdown workflow

`knitr::knit()` + Pandoc (+ LaTeX for PDF) = `rmarkdown::render()`



What is a {knitr} engine ?

Using R as example

```
```{r setup, include = FALSE}
knitr::opts_chunk$set(echo = FALSE)
```
```

```
```{<engine> <label>, <keys = values>}
<code content>
```
```

- `<engine>` defines **how** the `<code content>` will be processed,
- `<keys = values>` are **configurations** for the engine,
- `<label>` is the name of the chunk
  (equivalent to `label = <label>`).

There is more than just R !

Meet the other engines !

A subset of available engines

```
names(knitr::knit_engines$get())
```

```
##  [1] "awk"       "bash"      "coffee"    "gawk"      "groovy"    "haskell"
##  [7] "lein"      "mysql"     "node"      "octave"    "perl"      "psql"
## [13] "Rscript"   "ruby"      "sas"       "scala"     "sed"       "sh"
## [19] "stata"     "zsh"       "asis"      "asy"       "block"     "block2"
## [25] "bslib"     "c"         "cat"       "cc"        "comment"   "css"
## [31] "ditaa"     "dot"       "embed"     "exec"      "fortran"   "fortran95"
## [37] "go"        "highlight" "js"        "julia"     "python"    "R"
## [43] "Rcpp"      "sass"      "scss"      "sql"       "stan"      "targets"
## [49] "tikz"      "verbatim"  "glue"      "glue_sql"  "gluesql"   "upper"
## [55] "py"
```

Let's take a tour! 🚌

Processing chunk content as-is

Meet the `verbatim` engine

Include chunk content in a code block

.Rmd before knitting

````
Let's show an example of Rmd file content:

````{verbatim, lang = "markdown"}
We can output arbitrary content **verbatim**.

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

The content can contain inline code like
`r pi * 5^2`, too.
````
````

.md after knitting

````
Let's show an example of Rmd file content:

````markdown
We can output arbitrary content **verbatim**.

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

The content can contain inline code like
`r pi * 5^2`, too.
````
````

Processing chunk content as-is

Meet the embed engine

Include file content in a code block

.Rmd before knitting

````
```{embed, file = "macros.js"}
```
````

.md after knitting

````
```js
remark.macros.scale = function(w, alt="") {
  var url = this;
  return '<img src="' + url + '" style="width: ' + w + ';" alt="' + alt + '" />';
};
remark.macros.scaleh = function(h, alt="") {
  var url = this;
  return '<img src="' + url + '" style="height: ' + h + ';" alt="' + alt + '" />';

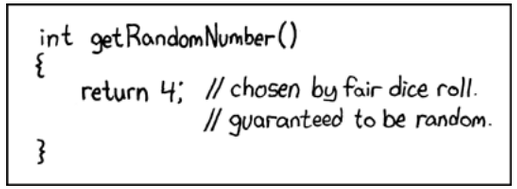
};
```
````

Processing chunk content as-is

Look at the `asis` engine

````
```{r}
getRandomNumber <- function() {
  sample(1:6, 1)
}
```

```{asis, echo = getRandomNumber() == 4}
According to https://xkcd.com/221/,
we just generated a **true** random number!
```
````



https://xkcd.com/221/

Processing chunk content as-is

Look at the `asis` engine

For `getRandomNumber() != 4`

````
```r
getRandomNumber <- function() {
  sample(1:6, 1)
}
```
````

For `getRandomNumber() == 4`

````
```r
getRandomNumber <- function() {
  sample(1:6, 1)
}
```

According to https://xkcd.com/221/,
we just generated a **true** random number!
````

Adding dependencies

Include CSS and JS easily in HTML

````
```{css, echo=FALSE}
/* Some CSS code that will will be included in
HTML document a `<style>` tag. */
```
````

````
```{js, echo=FALSE}
// some JS code that will be included in HTML
document in a`<script>` tag.
```
````

Useful to customize output directly from the Rmd file without external resource

Running other tools than R

Some built-in support

```{python}
import os
os.env
```

```{stata}
sysuse auto
summarize
```

```{bash}
ls *.Rmd | head -n 5
```

```{perl}
$test = "jello world";
$test =~ s/j/h/;
print $test
```

Chunk content is passed to the tools through `system2()`

Running other tools than R

Extend using `exec` engine

Using `node` CLI (which require `.js` extension for scripts)

````
```{exec, command='node', engine.opts = list(ext = ".js")}
function Display(x) { console.log(`Your number is ${x}`); }
Display(100);
```
````

````
```javascript
function Display(x) { console.log(`Your number is ${x}`); }
Display(100);
```

```
## Your number is 100
```
````

More in [knitr-examples repo](): [`124-exec-engine.Rmd`]()

Working with interoperability

More engines powered by other 📦

- Use the `sql` engine to run queries using **DBI** on compatible databases
- Use the `python` engine with **reticulate** to work seemlessly with R and Python chunks together
- Use the `scss` or `sass` engine to process a chunk content with **sass** package to insert a CSS in HTML
- Use the `bslib` engine to add rules to **bslib** themes withing Rmd

Extending {knitr} with custom engines

Any package can provide custom way to process chunk content

- **glue** has a `glue` engine to process chunk content as if passed to `glue::glue()` function
- **texPreview** has a `texpreview` engine to render TeX snippet from code chunk, in non-LaTeX output document
- **targets** offers a `targets` engine so that literate programming can be used to create a **targets** workflow (Target Markdown)
- **d3** has a `d3` engine where chunk content will be processed with **r2d3**

Extending {knitr} with custom engines

How to create a new engine ?

```
knitr::knit_engines$set(foo = function(options) {
  # the source code is in options$code; just do
  # whatever you want with it
})
```

- Use `knit_engines$set()` to register by name
- All knitr options are passed to the engine
- Code chunk content is in `options$code`

Extending {knitr} with custom engines

How to create a new engine ?

This engine will take the chunk content and make it upper case.

```
knitr::knit_engines$set(upper = function(options) {
  code <- paste(options$code, collapse = "\n")
  # Allow to hide result
  if (options$results == 'hide') return()
  # Allow to prevent processing
  if (options$eval) {}
    toupper(code)
  } else {
    code
  }
})
```

````
```{upper}
Hello, **knitr** engines!
```
````

```
HELLO, **KNITR** ENGINES!
```

Thank you !

https://cderv.rbind.io/slides/user2022-knitr-engines

https://github.com/cderv/user2022-knitr-engines


@cderv


@chrisderv