

# Literate Programming

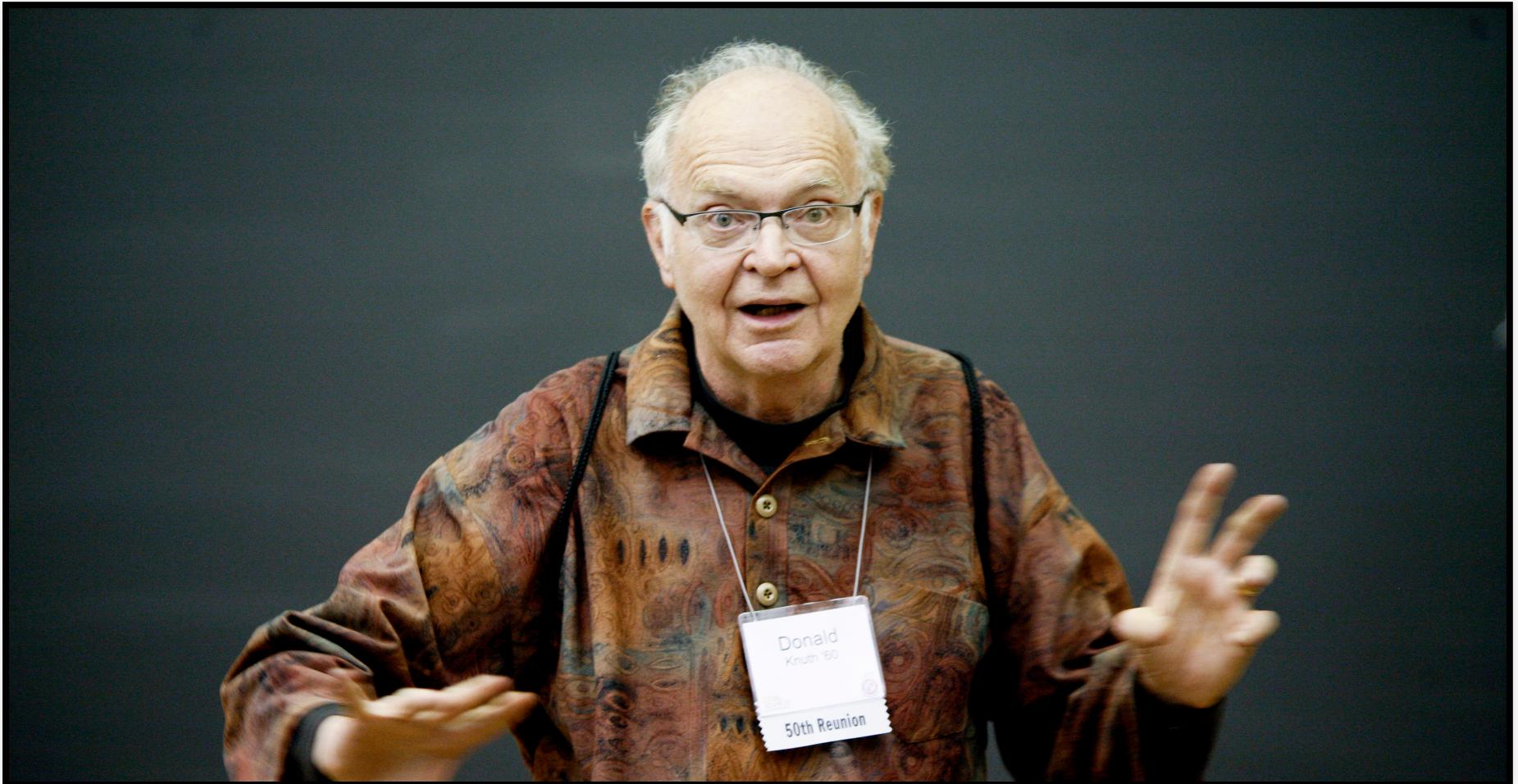
*why it matters, and how to make it easy*

Johan Hidding

netherlands eScience center

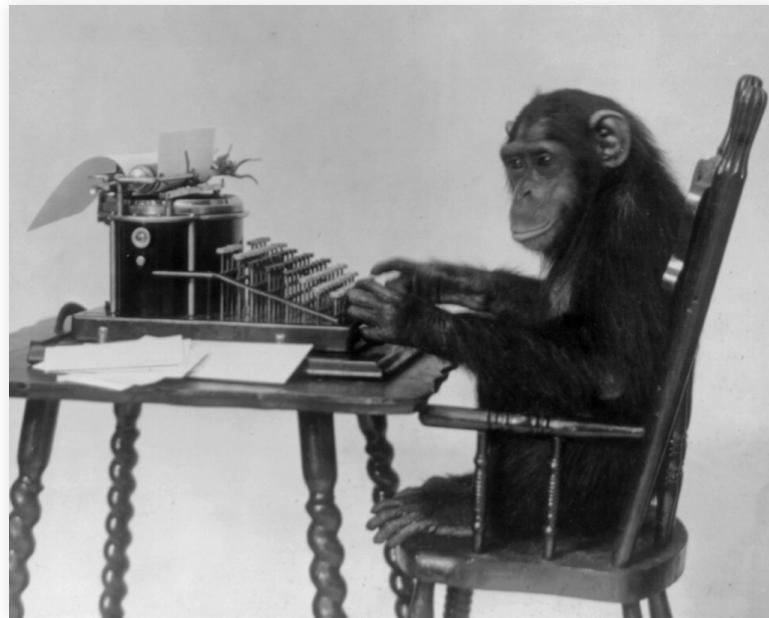
# Literate Programming

# Donald Knuth



Donald Knuth

# *Why Literate programming?*



*... nobody wants to admit writing an  
illiterate program ...*

# Motivation

**scientific rigour**

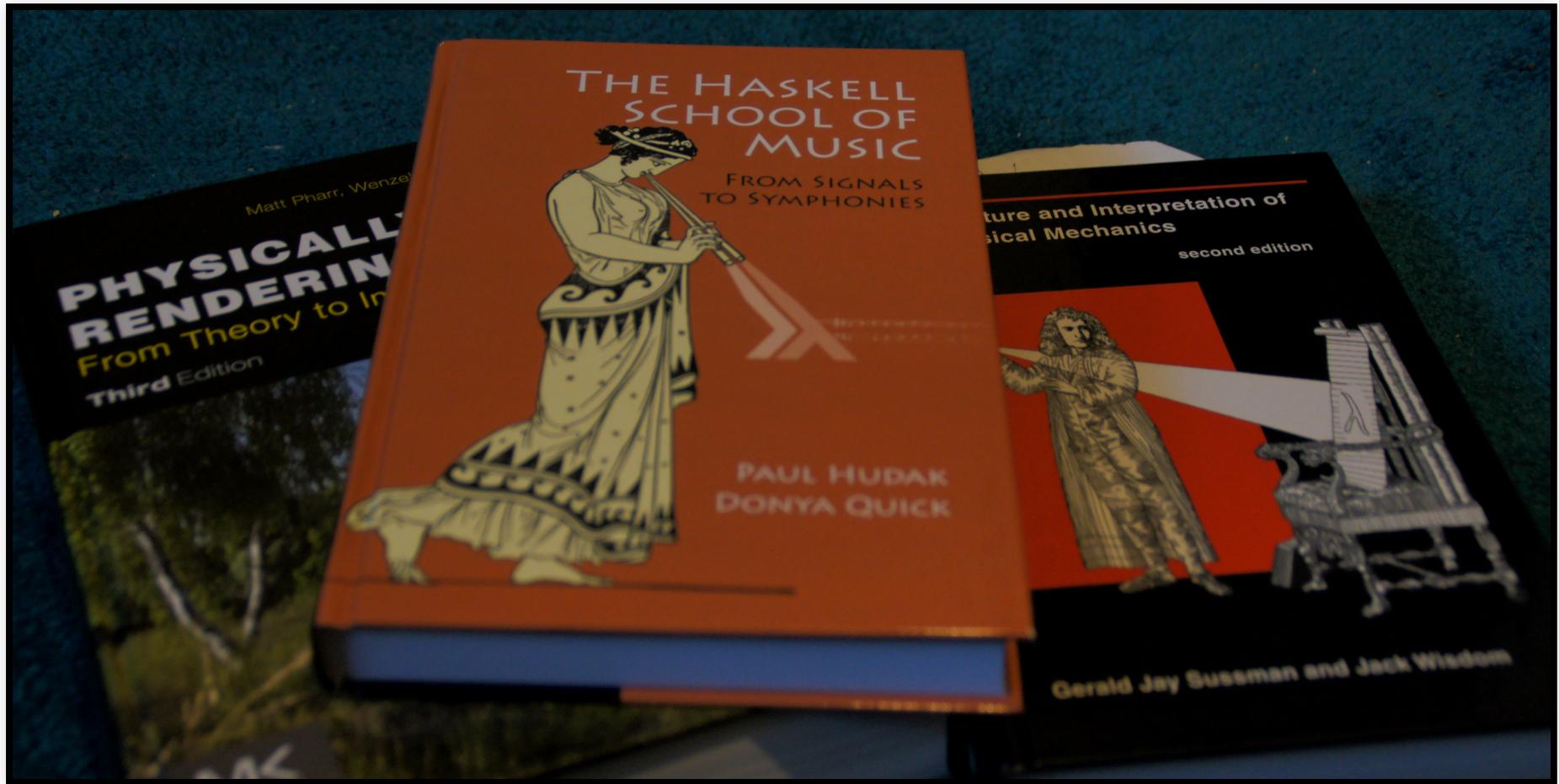
Should extend to **software** → “Open Science”

**epistemology**

Explain it to a six-year old  
**programming literacy**

Read other people’s code

# Books



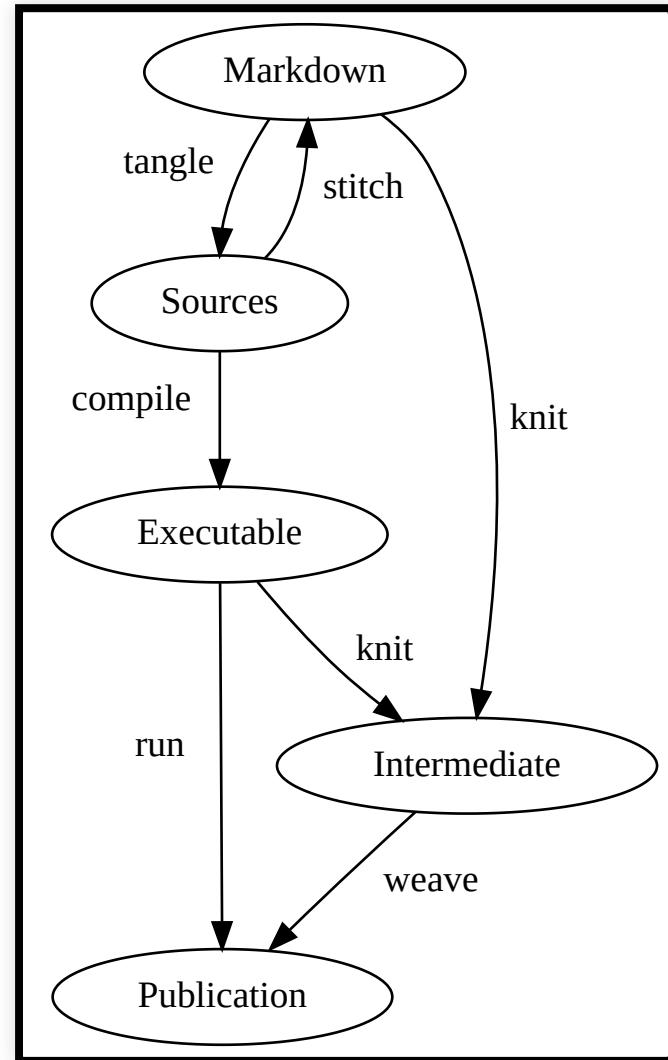
A sample of books using LP

# Existing tools

- Noweb
- Emacs Org-mode
- KnitR
- Jupyter

# Tools and workflow

- Pandoc
- Lua filters
- Entangled



# Hello, World!

# Introduction

- This example is written in a style of *literate programming* (Knuth 1984).
- We use a system of references known as *noweb* (Ramsey 1994).

# References

- References inside code blocks: << . . . >>

*file: «hello\_world.cc»=*

```
#include <cstdlib>
#include <iostream>

<<example-main-function>>
```

# References

- Code-blocks have names

«example-main-function»=

```
int main(int argc, char **argv) {  
    <<hello-world>>  
}
```

«hello-world»=

```
std::cout << "Hello, World!" << std::endl;
```

# References

- And can be appended to

«hello-world»=+

```
return EXIT_SUCCESS;
```

# Tangling

- Code blocks are *tangled* into source files.

file: «hello\_world.cc»=

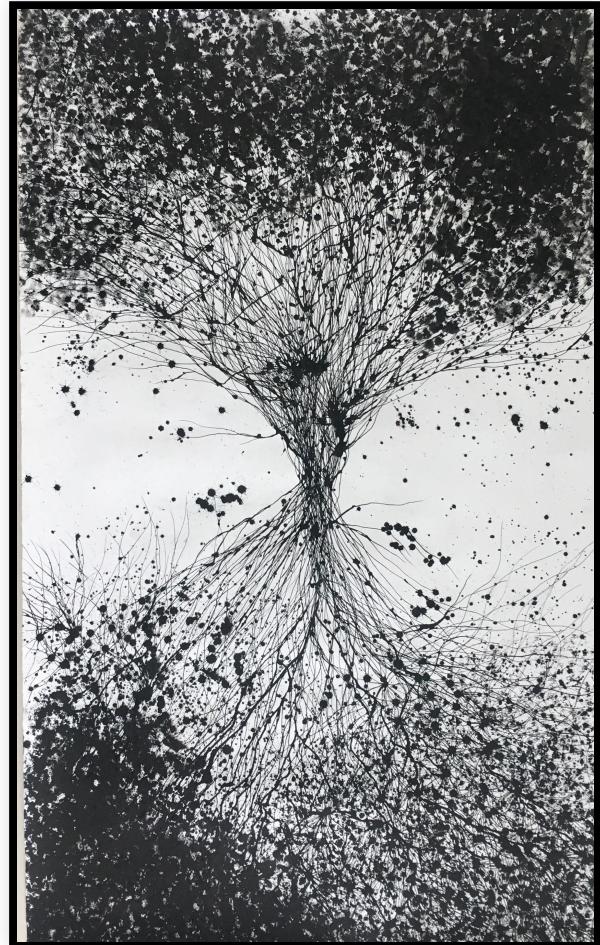
```
#include <cstdlib>
#include <iostream>

int main(int argc, char **argv) {
    std::cout << "Hello, World!" << std::endl;
    return EXIT_SUCCESS;
}
```

# Entangled

# Entangled

- Tangle and stitch
- Editor independent
- Language independent
- Demo



# Bibliography

Knuth, Donald Ervin. 1984. “Literate Programming.” *The Computer Journal* 27 (2): 97–111.

Ramsey, Norman. 1994. “Literate Programming Simplified.” *IEEE Software* 11 (5): 97–105.