Module 1 – lesson 02

In 1991, around the same time Jon Claerbout coined the term “reproducible research”, the computer scientist, Donald Knuth, introduced the concept of “literate programming.” The idea of literate programming is that software/computer programs are written in a language humans can understand rather than a language only machines can understand. In literate programming, computer code is embedded within the program's documentation as opposed to the documentation embedded within computer code; the code follows the structure of the documentation.

The program that Donald Knuth used to implement his idea of “literate programming” was called WEB, which he introduced in 1981. WEB linked the TeX typesetting or formatting system for creating documents with the Pascal computer programming language. WEB was one of the first systems to directly link documentation creation and typesetting with computer programming. Donald Knuth chose the name WEB because it implied a program of ideas pieced together from simple materials.

Since WEB was introduced, many other programs implementing literate programming have emerged. Here are a few to give you an idea of the variety available.

* CWEB also created by Donald Knuth with Silvio Levy which was adapted for the C and C++ compute language instead of Pascal
* Axiom developed by IBM
* Noweb
* Literate
* Funnel WEB
* Molly
* Codnar
* Jupyter Notebook (formerly IPython Notebook) and
* R Notebooks

So literate programming is an approach that moves away from writing computer programs in a high-level machine language and instead combines programming language with documentation language so that the program reads almost like an essay or a piece of literature. But what about dynamic documentation? Dynamic documentation allows for constant change and is a tool that provides up-to-date reports if certain components, such as data or analysis, change.

In 2002, Friederich Leisch, a statistics professor from the University of Natural Resources and Life Sciences in Vienna, released the SWEAVE program for dynamic documentation generation. Notably, SWEAVE allows R code to be embedded within LaTeX documents. LaTeX is a more modern version of the TeX typesetting program used by Donald Knuth. The really exciting feature of literate programming and dynamic documentation is highlighted in what Friedrich Leisch says about SWEAVE: since the underlying computer code is wholly integrated within the document itself, anytime there are changes to the underlying data or analyses or code, the report itself is automatically updated ON THE FLY!

The next evolution of ideas for literate programming and dynamic documentation have emerged from the R programming and RStudio communities. In 2012, Yihui Xie (yeewhay she) released the R package called knitr. This package was inspired by SWEAVE, and thus combines R code with text typesetting for producing documents. Like SWEAVE, knitr works with LaTeX but it also works with rmarkdown, which uses simple text markup syntax based on the original “markdown” package. The primary “markdown” package was introduced by John Gruber in 2004 to make it easier to “markup” plain text files for generating HTML documents – ideally without having to learn HTML. The rmarkdown package you’ll use throughout this course is built upon John Gruber’s “markdown”.

Rmarkdown itself was fully released in 2014 and its original objective was creating documents for the internet by creating HTML formatted documents. However, the “rmarkdown” package also leverages Pandoc for creating an even wider array of documentation formats – including:

* The DOC format, as used by Microsoft WORD or Google Docs
* The ODT format used by Libre Office
* The PDF format
* EPUB for electronic-books
* Slide shows using HTML5
* And the original TeX document formats and related TeX based slide formats like Beamer.

In this course, you won't interact directly with Pandoc, but it has been bundled with RStudio since 2015– so when you install RStudio, you'll also get the functionality of Pandoc. If you would like to learn more about Pandoc, you can visit their website at pandoc.org. Since Pandoc can convert many different document formats, it's often called the “swiss army knife” for document conversion. Pandoc is extremely versatile, allowing conversion between HTML web-based formats, word processor type formats, electronic publishing (or EPUB) formats, presentation slide-based formats, publication layout formats, TeX based formats, and many others.

Ultimately, the RStudio Interactive Development Environment becomes our central “HUB” for combining the capabilities of:

* The great packages of “knitr” and “rmarkdown”
* with the built-in functionality of Pandoc for document conversion
* plus the fantastic analysis and graphics capabilities of the R programming language.

From the RStudio interface, we can access all of this functionality and create documents on the fly in multiple formats for multiple end uses and products.