Investigación reproducible con Quarto y R

Felipe Ortega María Jesús Algar Emilio López Cano

2024-10-24

Tabla de contenidos

Preface

Este taller explica cómo utilizar Quarto, un software de creación de documentación científica, para crear publicaciones de calidad que integren contenido de texto formateado, gráficos, tablas, así como resultados de ejecución de código software en varios lenguajes, todo ello integrado en el propio documento.

Quarto se ha convertido en una herramienta muy versátil y potente en el conjunto de herramientas de los programadores científicos, en especial por proporcionar soporte para implementar buenas prácticas de **investigación reproducible**, incluyendo los principios FAIR. El intenso movimiento iniciado desde hace años por la comunidad científica para garantizar acceso en abierto no solo al producto final (e.g. una publicación) sino también a materiales adicionales (códig fuente, conjuntos de datos, figuras, procesos de trabajo, archivos de configuración, etc.) se ha convertido en un objetivo insoslayable para académicos y especialistas en muchos campos diferentes. Muchas publicaciones científicas de prestigio exigen ahora a los autores enviar estos materiales auxiliares junto con los borradores de sus manuscritos, para permitir que otros colegas reproduzcan y validen los resultados, repliquen sus estudios sobre nuevas cohortes de individuos o elementos o para contribuir a la interpretación de los resultados obtenidos.

Con la herramienta Quarto se puede combinar texto formateado (escrito en Markdown) junto con secciones de código ejecutable, todo integrado en un mismo documento. Estas secciones o *chunks* de código ejecutable pueden estar escritas en varios lenguajes: R, Python, Julia u Observable. Es incluso posible combinar en un mismo documento o colección de documentos secciones de código escritas en diferentes lenguajes de programación.

Este es un **taller práctico** que presenta ejemplos reales y comandos para crear paso a paso tus propios documentos con Quarto en poco tiempo. Además, junto a la explicación de los conceptos clave para entender este proceso también se ofrecen recomendaciones sobre buenas prácticas de trabajo, para guiar a los aprendices de Quarto en la dirección correcta.

Puedes aprender muchos más detalles sobre cuarto en la guía en línea https://quarto.org/docs/guide/. En particular, en https://quarto.org/docs/books se documenta en detalle cómo crear libros como este utilizando Quarto.

Parte I

Quarto

1 Documentos científicos

En su actividad diaria, los estudiantes, académicos y especialistas científicos producen gran cantidad de documentación de todo tipo: notas de laboratorio, apuntes, memorandos, informes técnicos y, sobre todo, artículos científicos para publicar sus descubrimientos y avances en un área de conocimiento. Normalmente, la creación de este tipo de documentos científicos conlleva una gran cantidad de tareas que involucran diferentes herramientas y posibles puntos de fallos.

La *figura X.X* muestra una descripción general esquemática de un proceso de trabajo clásico para la creación de documentos científicos. Con frecuencia, el elemento principal es un archivo maestro de procesador de textos (Word, OpenOffice/LibreOffice, etc.), una página web o un fichero LaTeX (si vamos a crear un documento PDF) que recoge todo el contenido.

Este archivo maestro se va llenando de contenido que procede de diversas fuentes, como por ejemplo:

- figuras y esquemas generados manualmente o mediante código software (como gráficos de visualización de datos);
- tablas y resúmenes que describen conjuntos de datos y resultados;
- resultados y evaluación del rendimiento de modelos o algoritmos; estadísticos o de aprendizaje automático;
- fórmulas y ecuaciones matemáticas;
- tablas de datos y otra información de utilidad;
- referencias bibliográficas (normalmente generadas con ayuda de algún programa de gestión de información bibliográfica).

Muchos de estos elementos fuerzan a los usuarios a ejecutar una y otra vez herramientas y programas externos, procedimientos y otras tareas para incorporar luego los nuevos resultados al fichero maestro. Debemos admitir que este proceso, en su mayor parte manual, además de tedioso puede ser muy propenso a que cometamos errores o descuidos. “¡Espera! He olvidado actualizar la figura 1”. “¿SEguro que estos son los últimos resultados de evaluación del modelo M?” “¿Has comprobado que hemos cargado la última versión del ficher de datos D?” Estas son preguntas comunes que surgen en el día a día de los equipos científicos.

Sin embargo, sería genial si no fuese necesario realizar todo ese proceso manual y, en ocasiones, muy frustrante de forma manual. ¿Tenemos alguna alternativa para evitarlo? Sí, la tenemos. La respuesta a nuestras necesidades nos la brinda un concepto muy poderoso: la **programación literaria.

1.1 Programación literaria

El concepto de programación literaria fue acuñado por el profesor Donald E. Knuth ya en 1984 Knuth (1984). Sí, no has leído mal, hace más de 40 años. Este concepto establece que debería ser posible integrar, en un solo documento científico, texto formateado y resultados de la ejecución de código software para componer dicho documento de forma dinámica. Entonces, ¿por qué hemos tardado tanto en poner en práctica esta idea? La visión de Knuth, aunque muy adelantada a su tiempo, era correcta, pero la tecnología de la época no permitía ponerla en práctica.

Sin embargo, hoy día contamos con todos los elementos indispensables para llevar esta idea a la práctica. Es más, contamos con una herramienta, Quarto, que nos va a permitir automatizar y gestionar todo el proceso de creación de documentos de programación literaria de forma rápida y fiable.

1.2 Investigación reproducible

1.3 Quarto para publicaciones científicas

Ahora que ya conocemos el concepto fundamental sobre el que se asienta el funcionamiento de Quarto y su aplicación para conseguir un mayor nivel de reproducibilidad y transparencia en nuestro proceso científico, vamos a explicar con más detalle el proceso que sigue Quarto para componer un documento. La figura X.X presenta un esquema con el proceso de creación del documento y los elementos y herramientas que entran en juego para conseguirlo.

- **Quarto engine**
- **Markdown** (contenido formateado):
- **Programming engine**: R, Python, Julia (lenguajes de programación que se pueden ejecutar en un entorno REPL).
- **Pandoc** (traductor universal de formatos documentales):
  - Primero se crea una salida en formato Markdown.
  - Después, usamos Pandoc para convertir el contenido Markdown en el tipo de salida seleccionado.
  - Opciones disponibles: HTML, PDF, Word.

1.4 Instalación de Quarto

https://quarto.org/docs/get-started/

Última versión disponible: `1.5.57`.

2 Tipos de documentos

En este capítulo, presentamos los principales tipos de documentos y colecciones de contenidos científicos que podemos generar con Quarto.

2.1 Documentos individuales

La forma más sencilla de trabajar con Quarto es crear un documento individual. Dicho documento podrá utilizar las secciones o *chunks* de código para leer datos de entrada o descargarlos de alguna fuente, procesarlos, analizarlos y mostrar los resultados. Se pueden añadir gráficos, tablas, ecuaciones, referencias bibliográficas y muchos otros elementos.

Los documentos tienen siempre una estructura estándar:

- *Preámbulo*: en el que se especifican opciones de configuración para la creación del documento con Quarto y sus herramientas asociadas.
- *Cuerpo*: la sección que alberga el contenido principal del documento, incluyendo secciones de texto formateado en Markdown y secciones de código ejecutable. El código software se podrá mostrar, si resulta de utilidad, o quedar oculto en el resultado final.
- *Referencias*: Al final del documento se incluyen las referencias bibliográficas, como es habitual en los textos científicos.

2.2 Libros

La evolución natural del caso anterior es reunir una colección de documentos individuales en un solo libro. *Quarto books* permite crear este tipo de documentos, estructurados en partes, capítulos y secciones. Las opciones de configuración permitirán confeccionar una portada de introducción para el sito web que contiene los capítulos (un documento por capítulo) o bien los elementos necesarios para crear un libro en PDF, semejante a los publicados por una editorial.

2.3 Artículos y publicaciones

Uno de los resultados clave en todo proceso científico es la producción de artículos y publicaciones (informes técnicos, etc.) que recojan los resultados y avances científicos conseguidos. En este caso, Quarto también nos podrá ayudar, con la colaboración de otros elementos indispensables como el paquete R `rticles`, que proporciona plantillas para generar artículos según las especificaciones de las principales publicaciones y editoriales científicas en multitud de campos de conocimiento.

2.4 Presentaciones

También es posible generar presentaciones (normalmente, en formato HTML) con diapositivas mediante Quarto. En este caso, tendríamos el soporte de varios paquetes y entornos de creación de presentaciones web a nuestra disposición, como `reveal.js` (HTML), Beamer (para LaTeX/PDF) o formato PPTX de MS Office.

Este caso no lo trataremos en este taller, pero se puede obtener más información en la guía online, disponible en https://quarto.org/docs/presentations/.

2.5 Sitios web

Otra opción que puede resultar interesante es crear sitios web personales (por ejemplo, para mostrar nuestro CV y una selección de trabajos destacados, publicaciones, etc.), blogs e incluso sitios web corporativos (organización, grupo de investigación) de forma rápida mediante Quarto. Existen numerosas plantillas gratuitas y de pago ya disponibles para crear sitios web con un aspecto armonizado, aunque necesitaremos aprender un poco de HTML y CSS para poder personalizar aún más nuestra web.

Aquí tenemos un ejemplo de sitio web de un investigador en tecnología medioambiental creado con Quarto: https://www.mm218.dev/. Más ejemplos de diferentes tipos de sitios web generados con Quarto: https://drganghe.github.io/quarto-academic-site-examples.html.

Se puede conseguir más información y tutoriales para crear sitios web con Quarto en https://quarto.org/docs/websites/.

2.6 *Dashboards*

Por último, es posible crear cuadros de mandos o *dashboards* personalizados para monitorización de datos, análisis de modelos y resultados o bien para ejemplos y aplicaciones docentes utilizando Quarto, tal y como se describen en la guía https://quarto.org/docs/dashboards/.

En este caso podemos incluir entre las herramientas Shiny, un paquete software para R (también disponible para Python) con el que crear aplicaciones interactivas basadas en datos de forma rápida y sencilla.

3 Proceso de trabajo

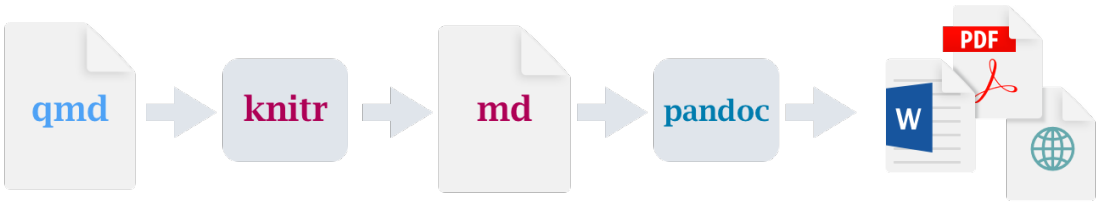

Figura 3.1: How it works. Fuente: RStudio.

3.1 Conjunto de herramientas

- Quarto
- Knitr
- Markdown
- Pandoc

Se necesitan programas para cargar los documentos de salida, según el formato: navegador web (HTML), MS Word (archivos DOC), visor PDF (archivos PDF).

3.2 Producir HTML

3.3 Producir PDF

3.3.1 Personalizar documentos PDF

Se pueden utilizar plantillas de documentos LaTeX predefinidas. Por defecto, Quarto utiliza varias plantillas de la colección de paquetes LaTeX koma-script.

4 Documentos individuales

La manera más sencilla de comenzar a utilizar Quarto es crear documentos individuales. Se trata de documentos autocontenidos, que incorporan texto formateado y código ejecutable en un único archivo.

Para crear un documento nuevo con Quarto, simplemente podemos usar las opciones del menu de RStudio o MS Visual Code, o bien crear un archivo con extensión `.qmd`.

4.1 Creación del documento con RStudio

Antes de empezar, comprueba que has instalado el software Quarto en tu máquina. Es un programa software independiente, que tiene que estar instalado para que el resto del proceso funcione (consulta la sección Sección 1.4).

Si ya tenemos instalada una versión reciente de RStudio, necesitaremos instalar los siguientes paquetes para el ejemplo:

```
install.packages("tidyverse")
install.packages("palmerpenguins")
install.packages("quarto")
```

Ahora, en RStudio creamos un nuevo proyecto eligiendo la opción *Quarto project*, tal y como aparece en la Figura 4.1.

Podemos nombrar el directorio de nuestro proyecto como `primer-ejemplo` y pulsamos `Create Project`.

Como resultado, nos debe aparecer un nuevo proyecto abierto en pantalla, con el aspecto que se muestra en la Figura 4.2.

En concreto, en el panel superior izquierdo podemos comprobar que, por defecto se ha abierto el editor *Visual*, que permite crear documentos Quarto de forma más intuitiva. Sin embargo, para empezar a familiarizarnos desde el principio con la estructura de un documento en quarto vamos a cambiar al editor *Source* para ver el código fuente, pulsando en el botón que se muestra en la figura Figura 4.3.

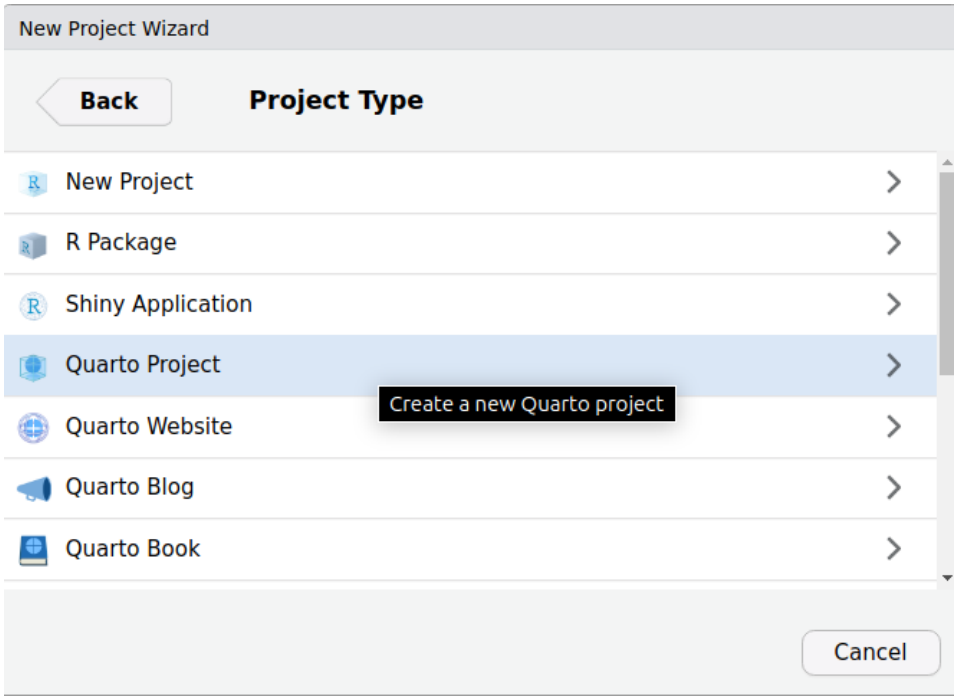

Figura 4.1: New Quarto project

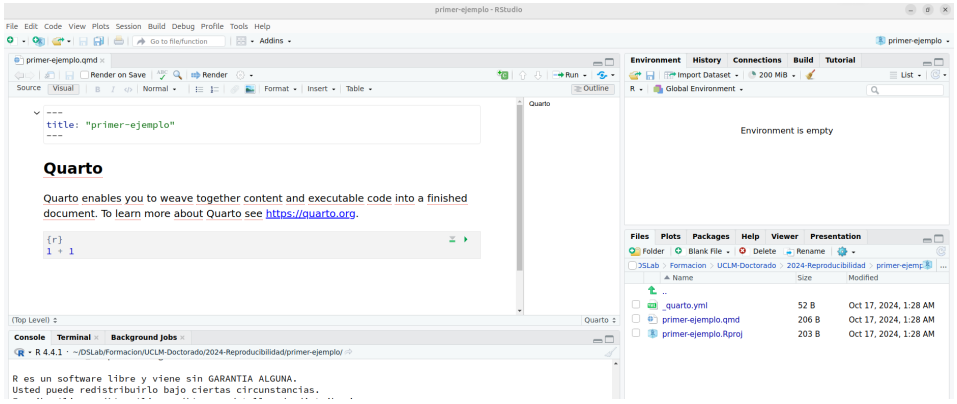

Figura 4.2: First example

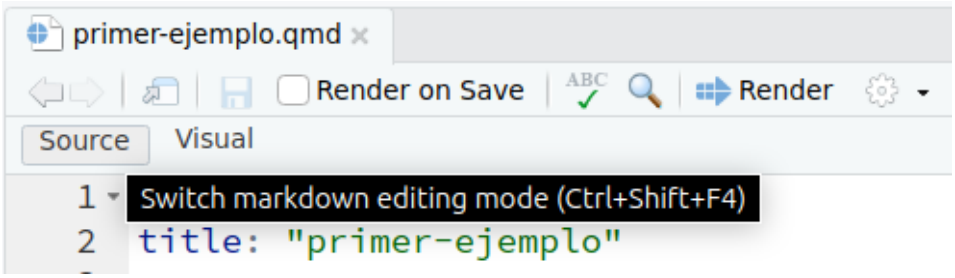

Figura 4.3: Source editor

4.2 Estructura del documento

La estructura de un documento individual en Quarto es esta.

```
---
title: "Mi primer documento"
author: John Doe
date: 2024-10-24
---

Aquí tenemos algo de contenido en formato **Markdown**.

```{r}
#| label: Etiqueta

1 + 1
```

Contenido *adicional* en Markdown.
```

El contenido del archivo consta de dos partes:

- **Preámbulo**: está delimitado por dos marcas `---`. Dentro de esta área podemos asignar valores a opciones de configuración para maquetar y crear el documento, tales como el título, autor/es, fecha, etc. También podemos configurar diversas opciones relacionadas con el formato de salida de los documentos.
- **Cuerpo del documento**: se compone de párrafos de texto formateados utilizando la sintaxis de marcado Markdown, que veremos después. Además, también se pueden intercalar en el texto secciones de código ejecutable o *chunks*, que se marcan siguiendo una sintaxis especial (como vemos en el ejemplo anterior).

Cada *chunk* de código ejecutable está delimitado de la siguiente manera

```{r}
# Código en R
```

Soporte para otros lenguajes de programación

Aunque en este taller nos centramos en el lenguaje R, debemos sabes que Quarto también soporta otros lenguajes de programación como Python, Julia u Observable.
Podemos cambiar el lenguaje de programación de cada *chunk* indicando su nombre al comienzo, como por ejemplo:

```{pyton}
# Código en Python
```

Sin embargo, para que pueda funcionar necesitaremos realizar algunas tareas adicionales de configuración.

4.2.1 El preámbulo

4.2.2 Listado de opciones

4.2.3 Sintaxis Markdown básica

4.3 Creación del documento (*output*)

4.3.1 Previsualización

4.3.2 Seleccionar el tipo de documento

4.3.3 Opciones básicas de configuración

4.4 *Chunks* de código ejecutable

https://quarto.org/docs/get-started/computations/

4.5 Herramientas para el autor

https://quarto.org/docs/get-started/authoring/

Parte II

Libros con Quarto

5 Libros

5.1 Herramientas de redacción

5.1.1 Figuras

5.1.2 Tablas

5.1.3 Ecuaciones

5.1.4 *Callouts*

5.2 Gestión de referencias

5.2.1 Referencias cruzadas en el documento

5.2.2 Referencias bibliográficas

5.3 Plantillas y personalización

6 Taller: colección de apuntes

6.1 Plantillas y herramientas

6.2 Gestión del proyecto

6.3 Publicación

Parte III

Publicaciones

7 Artículos y publicaciones científicas

7.1 Replicabilidad en publicaciones científicas

7.2 Figuras y gráficos para publicación

7.3 EL paquete `rticles`

7.4 Ejemplos y recomendaciones

8 Principios FAIR

8.1 Visión general

8.2 Publicación del código fuente

8.3 Publicación de conjuntos de datos

8.4 Gestión de referencias

DOI

Figshare

Zenodo

arXiv

etc.

9 Recursos adicionales

Listado de recursos adicionales de interés.

See Knuth (1984) for additional discussion of literate programming.

Referencias

Knuth, D. E. (1984). Literate Programming. *Comput. J.*, *27*(2), 97-111. https://doi.org/10.1093/comjnl/27.2.97

A Comandos de utilidad

A.1 Comandos Quarto

A.2 Celdas de código en R

B Entornos de desarrollo para Quarto

B.1 R Studio

B.2 Visual Studio

C Paquetes R de interés

C.1 `rticles`

Shiny

IOSlides, otras...

Otros

D Documentos PDF con LaTeX

D.1 Salida en formato PDF

D.2 Acerca de LaTeX

D.3 Ejemplos prácticos

Referencias

Knuth, D. E. (1984). Literate Programming. *Comput. J.*, *27*(2), 97-111. https://doi.org/10.1093/comjnl/27.2.97