

Informes automatizados: RMarkdown

Sitio: [Plataforma de Formación On line del Instituto Andaluz de
Administración Pública](#)
Curso: (I22F-PT05) Entorno de Programación R
Libro: Informes automatizados: RMarkdown

Imprimido por: ALFONSO LUIS MONTEJO RAEZ
Día: lunes, 18 de abril de 2022, 10:14

Tabla de contenidos

1. Introducción

2. Instalación

3. Crear un documento

4. Partes de un documento

4.1. Cabecera

4.2. Bloques de código. Chunk

4.3. Texto libre

5. Generar documento

1. Introducción



El paquete **rmarkdown** nos permite realizar documentos dinámicos con R usando el lenguaje Markdown.

Markdown es un lenguaje de marcación, similar a HTML, que permite dar formato a un texto de manera rápida y sencilla. Originalmente fue creado por John Gruber en lenguaje Perl pero se ha extendido a multitud de programas, entre ellos R.

En nuestro caso, vamos a ver una introducción a R Markdown, para conocer las principales órdenes y su potencialidad para la generación de informes automatizados.

2. Instalación

rmarkdown es un paquete que hay que instalar y cargar para que funcione ya que no viene por defecto en el paquete base de R.

La instalación se puede hacer de dos formas:

- **mediante órdenes.**
- **mediante asistente.**

Mediante órdenes

Lo haríamos usando **install.packages('rmarkdown', dependencies=T)** y a continuación usamos la orden **library(rmarkdown)** para cargar el paquete y poder trabajar con él.

Mediante asistente

Como para cualquier paquete de R, el programa RStudio nos ofrece la posibilidad de instalar y cargar un paquete desde su asistente.

Este proceso ya lo hemos explicado anteriormente en la Unidad 1. Para más detalles ver:

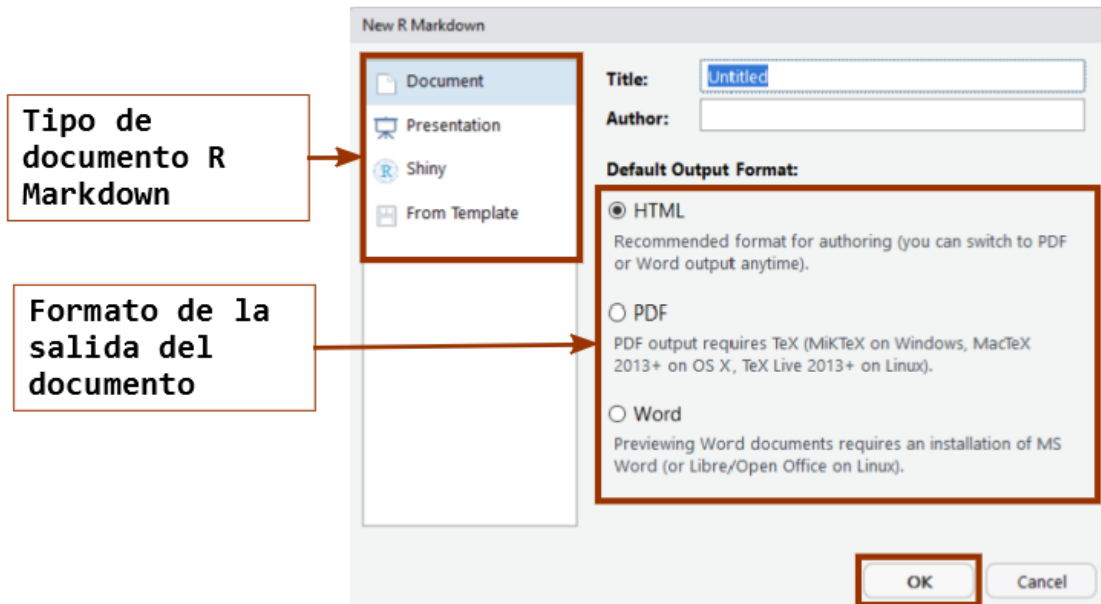
Unidad 1 → Empezar a trabajar con RStudio → 3. Gestión de paquetes → 3.3. Instalación y 3.4. Carga

3. Crear un documento

Para crear un documento Rmarkdown se puede hacer desde RStudio siguiendo la ruta:

File → New File → R Markdown

Una vez elegida esta opción aparece una pantalla donde podemos elegir las opciones de configuración.



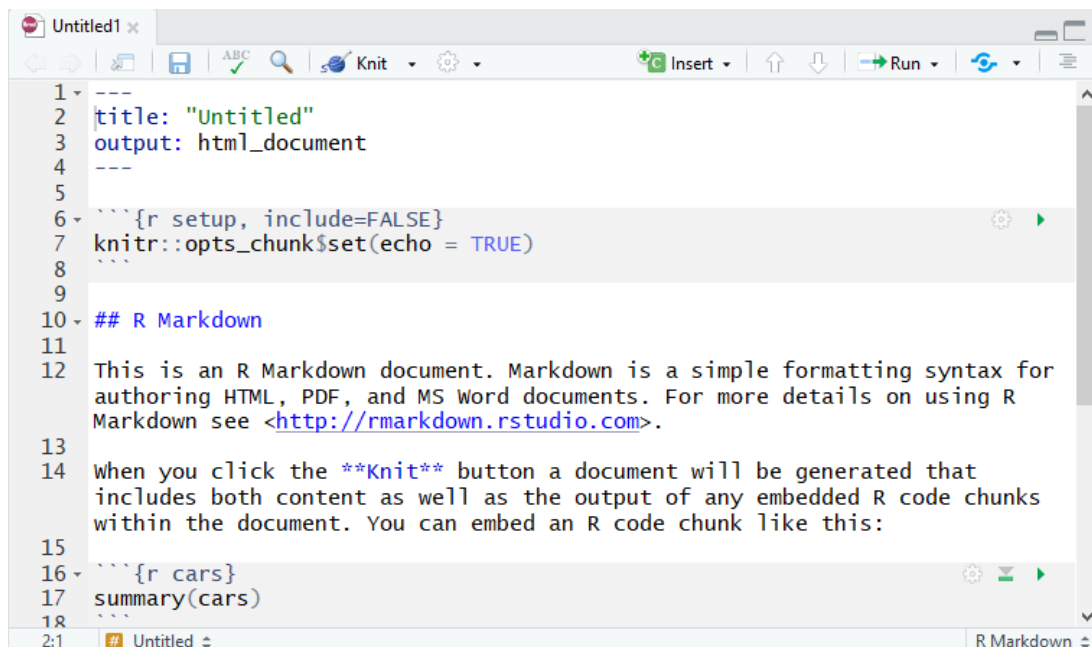
Las principales opciones que tenemos que elegir son:

- **Title.** Título del documento que se va a crear.
- **Tipo de documento.** Podemos elegir entre generar un documento, una presentación, incluir una aplicación Shiny o seleccionar una plantilla.
- **Tipo de salida.** Elegir el tipo de salida en el que se va a generar el documento. Las opciones son HTML, PDF o Word.

Lo más frecuente es generar un documento en formato HTML. Para generar un documento en PDF es necesario tener instalado un programa de transformación como Latex (programa de pago), Miktex (software libre), etc.

Para la descarga de Miktex os dejo el [enlace](#).

En nuestro caso, elegimos generar un Documento en formato HTML (lo que viene por defecto) y pulsamos el botón **OK**.



The screenshot shows the RStudio editor interface with a file named 'Untitled1'. The editor contains an R Markdown document template. The code is as follows:

```
1 ---
2 title: "Untitled"
3 output: html_document
4 ---
5
6 ```{r setup, include=FALSE}
7 knitr::opts_chunk$set(echo = TRUE)
8 ```
9
10 ## R Markdown
11
12 This is an R Markdown document. Markdown is a simple formatting syntax for
13 authoring HTML, PDF, and MS Word documents. For more details on using R
14 Markdown see <http://rmarkdown.rstudio.com>.
15
16 When you click the Knit button a document will be generated that
17 includes both content as well as the output of any embedded R code chunks
18 within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
```

The status bar at the bottom indicates the current file is 'Untitled' and the document type is 'R Markdown'.

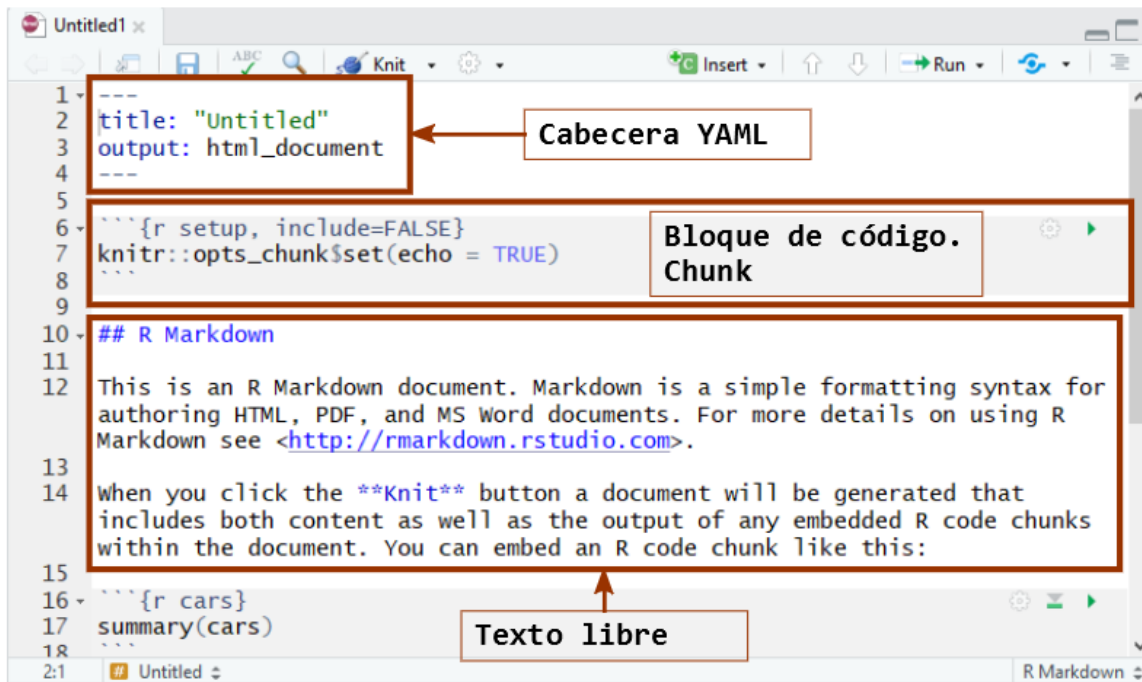
Esta acción genera un documento con varias partes que comentamos a continuación.

Cuando hacemos esto en RStudio, se genera automáticamente un documento Rmarkdown de ejemplo para poder trabajar con él.

4. Partes de un documento

La forma básica de un documento en R Markdown consta de tres partes:

- **Cabecera**, denominada YAML.
- **Bloques de código** llamados chunk.
- **Texto libre** que se ira insertando entre los bloques de código.



4.1. Cabecera

La **cabecera YAML** es la que contiene la información de cómo se va a generar el documento Markdown.

La parte del documento referida a la cabecera empieza y termina con tres guiones medios (---).

La inclusión de la cabecera en el documento es opcional pero aconsejable, ya que contiene información referida al tipo de salida que se va a generar, el autor, el título del documento, etc.

Esta cabecera se puede adaptar a cada tipo de documento que se va a generar y podemos especificar en ellas muchos otros parámetros de configuración.

```
---
output:
  pdf_document:
header-includes:
- \usepackage{caption}
- \captionsetup[table]{labelformat=empty}
- \captionsetup[figure]{labelformat=empty}
- \usepackage{fancyhdr}
classoption: landscape
fontfamily: mathpazo
fontsize: 11pt
params:
  far: ""
  mes: ""
  anno: ""
  unidad: ""
---
```

Por ejemplo, aquí podemos ver la **cabecera YAML** referente a un documento generado en PDF.

En ella se observa que se has especificado el tipo de letra, el tamaño de la fuente, el uso de parámetros y paquetes propios del programa Tex que estemos usando (por ejemplo, Miktex).

4.2. Bloques de código. Chunk

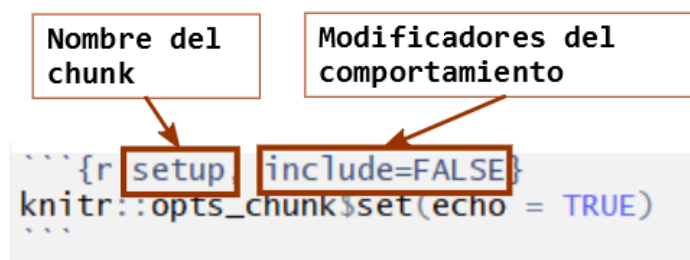
Los **chunks** son bloques de código R que generan resultados.

Para definir los bloques se inician con ````\{r\}` y finalizan con ````` (tilde grave).

Todo lo que se encuentra entre esos dos delimitadores serían las órdenes que forman un bloque (chunk) y son las que se van a ejecutar y van a aparecer en el informe que se genere.

En el chunk, podemos incluir las líneas de código R que queramos que se ejecuten en la generación del documento y también algunas opciones que son interesantes que nos ayudan a generar el documento Markdown.

Entre las cosas que podemos establecer dentro del chunk están el **nombre** y los **modificadores**. Estos irían entre las llaves (`{}`) que definen el chunk.



Asignarle un **nombre** a los chunk es importante (aunque no obligatorio), sobre todo cuando tenemos muchos y queremos buscarlos dentro de un documento muy extenso. Por supuesto, no puede haber dos chunk con el mismo nombre.

Otra cosa que podemos incluir son los **modificadores**.

El comportamiento del chunk cuando se genera el informe se puede controlar usando parámetros o modificadores que añadimos en el inicio del bloque de código.

Las principales opciones que podemos definir para variar la creación del informe son:

- **echo**. Muestra el código R en el documento final. Posibles valores: TRUE o FALSE.
- **eval**. Evaluación del código del bloque. Posibles valores: TRUE o FALSE.
- **include**. Muestra el código y el resultado en el documento final. Posibles valores: TRUE o FALSE.
- **message**. Modificar la aparición de los mensajes durante la ejecución del código. Posibles valores: TRUE o FALSE.
- **warning**. Modificar la aparición de los avisos durante la ejecución del código. Posibles valores: TRUE o FALSE.
- **results**. Modifica la aparición de los resultados en el informe. Posibles valores:
 - **hide**. Oculta los resultados.
 - **markup**. Muestra el resultado del código (por defecto).
 - **asis**. Muestra los resultados línea a línea de manera literal.
 - **hold**. Muestra el resultado del código de golpe al final del bloque.

Los parámetros se pueden usar de manera conjunta separándolos con comas.

Ejemplo

El parámetro **echo** sirve para mostrar/ocultar el código en el informe. Por defecto, siempre muestra el código.

Código ejecutado

```
```{r, echo=FALSE}
a<-5
a+1
sqrt(5)
```
```

Resultado

```
## [1] 6
```

```
## [1] 2.236068
```

FALSE o F. Código oculto, solo muestra resultados

```
```{r, echo=TRUE}
a<-5
a+1
sqrt(5)
```
```

```
a<-5
a+1
```

```
## [1] 6
```

```
sqrt(5)
```

```
## [1] 2.236068
```

TRUE o T. Muestra código y resultados (valor por defecto)

El parámetro **eval** sirve para mostrar/ocultar el resultado del código en el informe. Por defecto, siempre muestra el resultado.

Código ejecutado

```
```{r, eval=FALSE}
a<-5
a+1
sqrt(5)
```
```

Resultado

```
a<-5
a+1
sqrt(5)
```

FALSE o F. Resultados ocultos, solo muestra código

```
```{r, eval=TRUE}
a<-5
a+1
sqrt(5)
```
```

```
a<-5
a+1
```

```
## [1] 6
```

```
sqrt(5)
```

```
## [1] 2.236068
```

TRUE o T. Muestra código y resultados (valor por defecto)

El parámetro **results** sirve para modificar como se muestran el código y resultado.

Código ejecutado

```
```{r, results="asis"}  
a<-5
a+1
sqrt(5)
```
```

```
```{r, results="hold"}  
a<-5
a+1
sqrt(5)
```
```

```
```{r, results="hide"}  
a<-5
a+1
sqrt(5)
```
```

Resultado

```
a<-5  
a+1
```

```
[1] 6
```

```
sqrt(5)
```

```
[1] 2.236068
```

asis. Código y resultados intercalados

```
a<-5  
a+1  
sqrt(5)
```

```
## [1] 6  
## [1] 2.236068
```

hold. Primero las órdenes después el resultado

```
a<-5  
a+1  
sqrt(5)
```

hide. Solo las órdenes. El resultado no aparece

4.3. Texto libre

El otro componente de un R Markdown es el **texto** que podemos intercalar entre los bloques de código.

El texto lo podemos modificar y editarlo para añadirle efectos que mejoren la visualización del documento final.

Estas modificaciones las añadimos con marcas junto al texto. Aquí van algunas de las más usadas.

- **Negrita.** ****Texto****
- **Cursiva.** **Texto**
- **Enlace web.** [\[link\]\(www.texto.com\)](#)
- **Superíndice.** Texto^{2^}
- **Título:** ## Texto (cuantos más almohadillas menor tamaño de letra)

También podemos añadir en el texto cálculos y órdenes de R para que se ejecuten y generen un resultado en el informe. Para hacerlo tenemos que usar ``r orden``.

Por ejemplo, si queremos escribir un texto donde se calcule una media, tenemos que usar ``r mean(Datos$variable)``

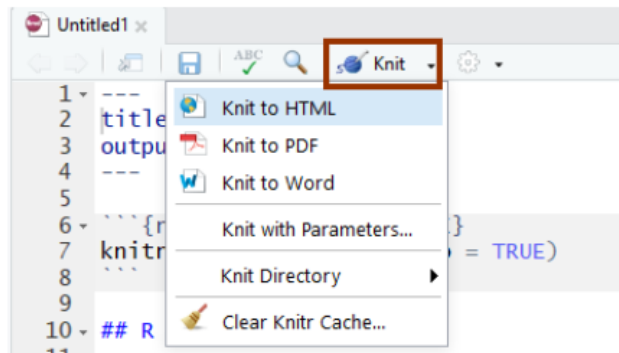
Esto nos permite incluir cálculos en los informes que se van actualizando de manera automática al cambiar los datos de origen.

Estos aspectos le dan mucha potencia a R Markdown para generar informes o documentos que incorporan gráficos, tablas o textos explicativos con datos.

5. Generar documento

Una vez hemos configurado todas las partes del documento, tenemos que compilarlo para ver el resultado final.

Para ello, tenemos que irnos al botón **Knit** de RStudio.



Desde esta opción podemos elegir el formato en el que se va a generar el documento final. En nuestro caso elegimos **Knit to HTML**.

Si no tenemos salvado el código, nos aparecerá un ventana donde nos pide que elijamos la ubicación donde se va a guardar el código Rmarkdown y el archivo HTML generado. Los archivos con código Rmarkdown tienen como extensión **.Rmd**

Untitled

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.


When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

summary(cars)

| ## | speed | dist |
|------------|-------|----------------|
| ## Min. | : 4.0 | Min. : 2.00 |
| ## 1st Qu. | :12.0 | 1st Qu.: 26.00 |
| ## Median | :15.0 | Median : 36.00 |
| ## Mean | :15.4 | Mean : 42.98 |
| ## 3rd Qu. | :19.0 | 3rd Qu.: 56.00 |
| ## Max. | :25.0 | Max. :120.00 |

Including Plots

You can also embed plots, for example:



Texto libre formateado

Código del chunk

Resultado del chunk

Texto libre formateado

Resultado del chunk con el código oculto

Si generamos el código que RStudio ha creado por defecto, podemos observar el documento HTML final donde se incluye un texto formateado para que aparezcan títulos, enlaces, palabras en negrita y varios chunks con su resultado. Un gráfico y una tabla con un resumen estadístico del conjunto de datos cars (que está por defecto en R).

Ejemplo 1

Vamos a formatear un texto que incluya un enlace a la web de la Junta de Andalucía y muestre un gráfico de barras.

El documento R Markdown quedaría así:

```
# Informe de análisis
## Detalle provincial

En este apartado se va a incluir un gráfico con la información del número de municipios por prov
Esta información se ha extraído de [www.juntadeandalucia.es].

```{r}
with(Datos_Demograficos, Barplot(Prov_Factor, xlab="Prov_Factor", ylab="Frequency"))
```
```

Y el resultado al generar el informe en HTML quedaría así:

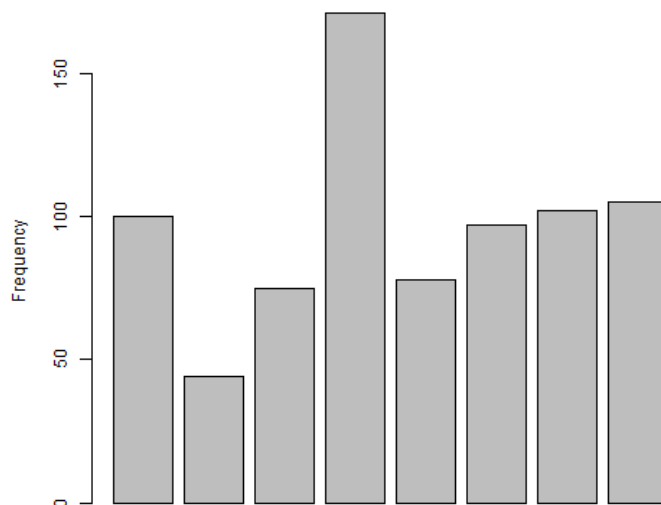
Informe de análisis

Detalle provincial

En este apartado se va a incluir un *gráfico* con la información del **número de municipios** por provincias.

Esta información se ha extraído de [www.juntadeandalucia.es].

```
with(Datos_Demograficos, Barplot(Prov_Factor, xlab="Prov_Factor", ylab="Frequency"))
```



Ejemplo 2

Al informe anterior, queremos añadirle en el texto el nombre y la población de la ciudad con más habitantes de Andalucía.

El documento R Markdown quedaría así:

```
# Informe de análisis
## Detalle provincial

El municipio con más población de Andalucía es `r Datos_Demograficos[which.max(Datos_Demograficos$Poblacion)]` habitantes.

```{r}
with(Datos_Demograficos, Barplot(Prov_Factor, xlab="Prov_Factor", ylab="Frequency"))
```
```

Código R. Empieza `r` y termina con `

Y el resultado al generar el informe en HTML quedaría así:

Informe de análisis

Detalle provincial

El municipio con más población de Andalucía es Sevilla (capital) con 689434 habitantes.

```
with(Datos_Demograficos, Barplot(Prov_Factor, xlab="Prov_Factor", ylab="Frequency"))
```

El nombre de la ciudad y la población se han calculado del data frame de forma automática. Si se repite el informe con los datos del año siguiente, el número de habitantes se actualizaría solo, sin necesidad de escribirlo.