4
Files and Documents

In your work with data, you will be using and creating computer files of various sorts. Of particular importance are three basic roles for files:

1. storing data tables
2. listing R instructions
3. writing reports with narrative, graphics and conclusions

Different file types are used for each of these roles. These types can be referred to by the filename extension of the files.

Table 4.1: Common file types and their uses

File Role	Common file types	Software
Data storage	`.csv`, `.xlsx`, etc.	Spreadsheets
R instructions	`.Rmd`	Text editor, compiler
End reports	`.html`, `.pdf`, `.docx`, etc.	web browser, word processor

Filename Extensions

The *filename extension* is one or more letters following the last period in the name: `.xlsx`, `.docx`, `.html`, `.mpeg`. (Other widely used extensions are `.pdf`, `.zip`, `.png`.) These letters indicate the format of the file and often set which program will be used to open the file: a spreadsheet, a word processor, a browser, or a music player in the examples. Or, in plainer language, the filename extension tells you the *kind* of file.

FILENAME EXTENSION: The letters following the last period in a file name. This suffix indicates the file type, that is, what software to use to interpret the file.

If you are looking at files through RStudio, the filename extension will always be displayed. If you are using your your own computer's file browser, your system may have been set up to hide the extension.

When referring to files within R statements or an Rmd file, you must **always** include the filename extension.

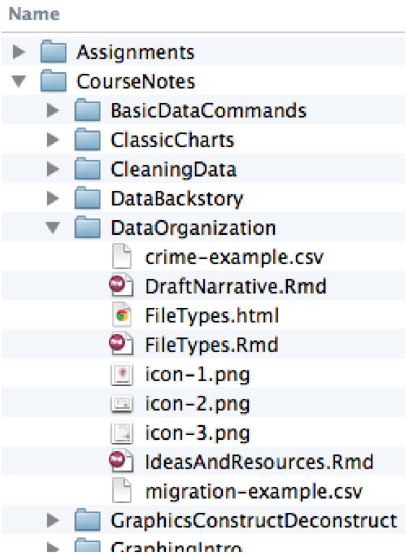
Paths

A filename is analogous to a person's first name. Just as first names are unique within a nuclear family, so filenames must be unique within a folder or directory.

You are probably used to seeing folders and the files they contain organized on your computer as in Figure 4.1. The *file path* is the set of successive folders that bring you to the file.

To run on with the family metaphor ... You are identified within a household by your first name. The path would tell you which specific nuclear family you belong to, perhaps in the form of your address, like this: *USA/Saint Paul/55105/703 Lincoln Avenue*.

FILE PATH: Information that specifies the location of a file in your file system.



```
Name
▶ Assignments
▼ CourseNotes
    ▶ BasicDataCommands
    ▶ ClassicCharts
    ▶ CleaningData
    ▶ DataBackstory
    ▼ DataOrganization
          crime-example.csv
          DraftNarrative.Rmd
          FileTypes.html
          FileTypes.Rmd
          icon-1.png
          icon-2.png
          icon-3.png
          IdeasAndResources.Rmd
          migration-example.csv
    ▶ GraphicsConstructDeconstruct
    ▶ GraphingIntro
```

Figure 4.1: Folders contained within folders, as shown by a file browser on Apple OS-X.

There is a standard format for file paths. An example:

```
/Users/kaplan/Downloads/0021_001.pdf
```

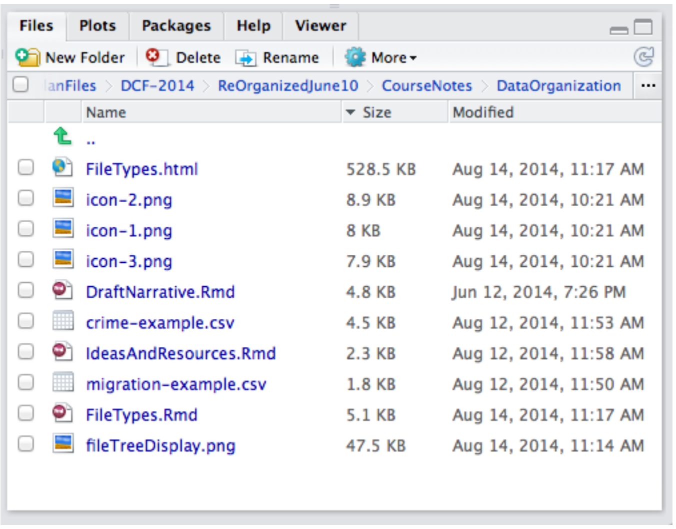
Here the filename is `0021_001`, the filename extension is `.pdf`, and the file itself is in the `Downloads` folder contained in the `kaplan` folder, which is in turn contained in the `Users` folder. The starting / means "on this computer".

The R `file.choose()` — which should be used **only in the console**, not in an Rmd file — brings up an interactive file browser. You can select a file with the browser. The returned value will be a quoted character string with the path name.

```
file.choose() # then select a file
```

```
[1] "/Users/kaplan/Downloads/0021_001.pdf"
```

In RStudio, the *Files* tab will display the path near the top. In Figure 4.2, the ten files lised are all in the same folder, whose path ends with\ `DCF-2014/ReOrganizedJune10/CourseNotes/DataOrganization`.

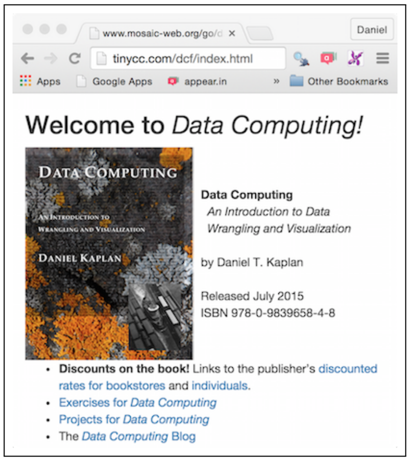


anFiles > DCF-2014 > ReOrganizedJune10 > CourseNotes > DataOrganization

Name	Size	Modified
..		
FileTypes.html	528.5 KB	Aug 14, 2014, 11:17 AM
icon-2.png	8.9 KB	Aug 14, 2014, 10:21 AM
icon-1.png	8 KB	Aug 14, 2014, 10:21 AM
icon-3.png	7.9 KB	Aug 14, 2014, 10:21 AM
DraftNarrative.Rmd	4.8 KB	Jun 12, 2014, 7:26 PM
crime-example.csv	4.5 KB	Aug 12, 2014, 11:53 AM
IdeasAndResources.Rmd	2.3 KB	Aug 12, 2014, 11:58 AM
migration-example.csv	1.8 KB	Aug 12, 2014, 11:50 AM
FileTypes.Rmd	5.1 KB	Aug 14, 2014, 11:17 AM
fileTreeDisplay.png	47.5 KB	Aug 14, 2014, 11:14 AM

Figure 4.2: Filenames and their file path shown in the Files tab in RStudio. Only part of the path is shown: the folders closest to the files.

QUOTES OR NOT FOR A FILE PATH? When you are referring to a file path in R, it will always be in quotation marks; it's a character string. Other software such as web browsers don't use quotes when specifying a path.



Welcome to *Data Computing!*

Data Computing
An Introduction to Data
Wrangling and Visualization

by Daniel T. Kaplan

Released July 2015
ISBN 978-0-9839658-4-8

- **Discounts on the book!** Links to the publisher's discounted rates for bookstores and individuals.
- Exercises for *Data Computing*
- Projects for *Data Computing*
- The *Data Computing* Blog

Figure 4.3: A browser directed to the URL `http://tiny.cc/dcf/index.html`
.

URLs

You have probably noticed URLs in the locator window near the top of your browser. In Figure 4.3, the URL is:
`http://tiny.cc/dcf/index.html`.

A URL includes in its path name the location of the server on which the file is stored (e.g. `tiny.cc`) followed by the path to the file on that surver. Here, the path is `/dcf` and the file is `index.html`.

You will sometimes need to copy URLs into your work in R, to access a dataset, to make a link to some reference, etc. Remember to copy the entire URL, including the `http://` part if it is there.

Some common filename extensions for the sort of web resources you will be using:

- `.png` for pictures
- `.jpg` or `.jpeg` for photographs
- `.csv` or `.Rdata` for data files
- `.Rmd` for the human editable text of a document
- `.html` for web pages themselves

Data Files

Data tables are often stored individually as files. There are many formats for data files. Among the most common is the `.csv` spreadsheet format, popular because reading it is a standard feature of many data analysis packages (including R).

If you use R extensively, you will also encounter data in `.Rda` (or `.Rdata`), an efficient format for storing data and other information specifically for R. When you get data from an R package, like this:

```
data(CPS85, package="mosaicData")
```

you are in fact reading in an `.Rda` file associated with the package.

There are many other formats for files containing data tables or the information needed to put the contents in data-table format. These are discussed in Chapter 15. Increasingly, data are accessed through *database* systems. In such a case, rather than reading the database as a whole, you make "queries" to access specific data you need.

Files containing data tables are often distributed via the web. These files are not any different than files on your own computer.

DATABASE: A computer system used to store, update, and access data. Relational data bases consist of data tables.

You access such files through a Uniform Resource Locator, better known as a *URL*. For instance, on the *Data Computing* web site, there are a number of `.csv` files. You can access them from R with commands like this:

URL: A name for a specific resource, such as a file, on the Internet.

```
Engines <- read.file("http://tiny.cc/mosaic/engines.csv")
Engines
```

Engine	mass	ncylinder	strokes	displacement	bore	stroke	BHP	RPM
Webra Speedy	0.14	1	2	1.80	13.50	12.50	0.45	22000
Motori Cipolla	0.15	1	2	2.50	15.00	14.00	1.00	26000
Webra Speed 20	0.25	1	2	3.40	16.50	16.00	0.78	22000
... and so on for 39 rows								

So far as accessing data is concerned, there's nothing fundamentally different in reading a file from a URL than reading a file on your own computer.

Documenting your work with Rmd files

The purpose of data wrangling and visualization is communication: condensing and presenting data in a form that conveys information. An important part of communication is documentation and reporting.

Writing is not a linear process. Ideas are presented, revised or abandoned, corrected, re-focussed, and re-arranged. Data-oriented technical reports tie together narrative, graphics and summary tables and are based on potentially complex computer commands. There is an interplay between the computer commands and the narrative. Results from the computer may drive reconsideration of the narrative. Gaps in the narrative may point to shortcomings or omissions in the computer commands. And, always, there is the possibility of errors in writing commands and the need to document commands so that they can be checked and corrected. As well, data are commonly updated, corrected or extended.

The familiar practice of cutting and pasting from the computer console into a word processor does not address these features of technical reports. Cutting and pasting makes it hard to revise or update a report; you've got to cut out the old and paste in the new, figuring out for yourself which is which. This introduces the likelihood of error. And, there's nothing to document the linkages between the computer commands and the word-processed document.

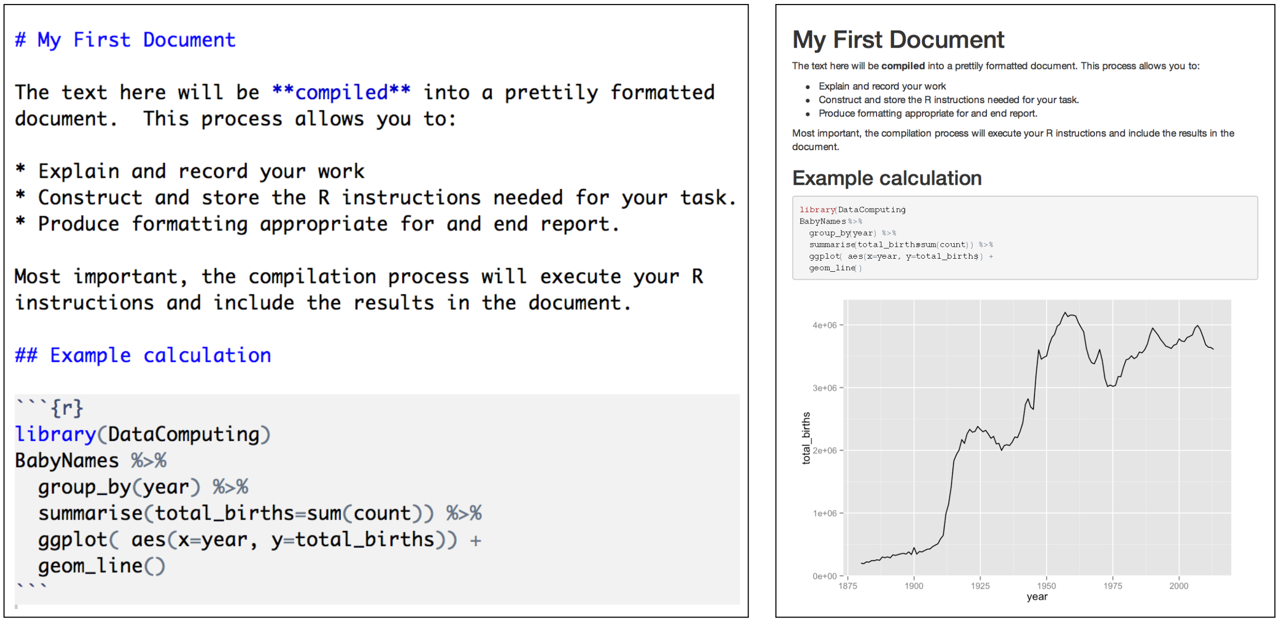
An important concept in data-driven reporting is "reproducibility." The idea is to be able to reproduce your entire document without any manual intervention, and, more important, to be easily able to generate a new report in response to changes in data or revisions in computer commands. In other words, reproducible reports contain all the information needed to generate a new report. Common document formats such as `.pdf`, `.docx`, or `.html` do not offer support for reproducibility.

In R, reproducible reporting is provided by the `.Rmd` file format and related software. An `.Rmd` file integrates computer commands into the narrative so that, for instance, graphics are produced by the commands rather than being inserted from another source.

YOU CREATE AND MODIFY REPRODUCIBLE REPORTS within RStudio's text *editor*. They contain ordinary text and punctuation: no formatting, color, images, etc. Instead, you use the `.Rmd` file to describe, using ordinary characters, both what you want the eventual format to look like and what R commands you want the computer to carry out in generating the report.

EDITOR: A program for constructing pure-text documents, such as R instructions and Rmd files

Figure 4.4 shows a simple `.Rmd` document, something you might write. The figure also shows the `.html` file that is the result of compiling the `.Rmd`.



~~~
# My First Document

The text here will be **compiled** into a prettily formatted
document.  This process allows you to:

* Explain and record your work
* Construct and store the R instructions needed for your task.
* Produce formatting appropriate for and end report.

Most important, the compilation process will execute your R
instructions and include the results in the document.

## Example calculation

```{r}
library(DataComputing)
BabyNames %>%
  group_by(year) %>%
  summarise(total_births=sum(count)) %>%
  ggplot( aes(x=year, y=total_births)) +
  geom_line()
```
~~~

My First Document

The text here will be **compiled** into a prettily formatted document. This process allows you to:

- Explain and record your work
- Construct and store the R instructions needed for your task.
- Produce formatting appropriate for and end report.

Most important, the compilation process will execute your R instructions and include the results in the document.

Example calculation

```
library(DataComputing)
BabyNames %>%
  group_by(year) %>%
  summarise(total_births=sum(count)) %>%
  ggplot( aes(x=year, y=total_births)) +
  geom_line()
```

Figure 4.4: The `.Rmd` file on the left consists of text. It is automatically compiled to the formatted `.html` document, with the results of calculations, shown on the right.

The Write/Compile Cycle

After you edit an `.Rmd` file, you *compile* the report into a reader-friendly document format such as `.pdf`, `.docx`, or `.html` that can easily be printed or displayed on the report-reader's computer. You

COMPILE: The process of a computer translating a file from one format to another.

never edit the reader-friendly document; it's created automatically from your `.Rmd` instructions. When you want to update or modify or correct the end report, you edit the `.Rmd` file.

It's a good strategy to compile your `.Rmd` frequently. Start with a small, simple document. Then add a bit more to it: one paragraph or one "chunk" (see below). Compile again. If something goes wrong, you will have a good idea of where the problem lies. Go back and fix things. Make small changes, compile, see if it worked. Repeat. Figure 4.5 shows several steps in such an editing cycle.

It's impossible to avoid errors; even professionals make them. Instead, adopt a process that let's you identify errors quickly so that you can fix them before moving on. The shorter the write/compile cycle, the easier it will be to know when you have erred.

Interactive commands such as `file.choose()` or `scatterGraphHelper()` cannot be used in an an `.Rmd` file; there's no opportunity for interaction in the compilation process. Instead, use the interactive command in the console, get the result, and paste that result into your `.Rmd` file.

Command "Chunks"

The R commands in an `.Rmd` file go into *chunk*s, a range of lines in the documents that are delimited in a special way so that they will be executed as part of the `.Rmd` → `.html` compilation process.[1] The opening delimiter is `` ```{r} ``. The closing delimiter is simply `` ``` ``. For example:

CHUNK: A region in an `.Rmd` file that contains R statements to be executed when the `.Rmd` is compiled.

~~~
```{r}
Saltpeter <- read.file("http://tiny.cc/DCF/saltpeter.csv")
```
~~~

Both the opening and closing delimiters for a chunk are "back-quotes," a quote character that goes from upper-left to bottom-right. On many keyboards, back-quote is on the same key as tilde (~), like this:

Most .Rmd files will draw on a library that needs to be loaded into the R session. When you compile `.Rmd` → `.html`, R starts a brand new session that is, initially, empty and with no libraries loaded. When the compilation is complete, that session evaporates, leaving as its only residue the `.html` result.

Often, the first chunk in your document will be an instruction to load one or more libraries. Since this will be in just about every `.Rmd` document, it can be called the *boilerplate* chunk. It looks like this: A boilerplate chunk goes at the start of the document. It loads the libraries that the following commands will use.

BOILERPLATE: A standardized piece of text intended for re-use in many documents.

~~~
```{r include=FALSE}
library(DataComputing)
# and any others that you need, e.g.
library(mosaic)
```
~~~

The `include=FALSE` chunk argument helps to prettify the document: it is an instruction not to show the contents of the chunk in the output file.

[1] Exactly the same applies when compiling to `.pdf` of `.docx`.

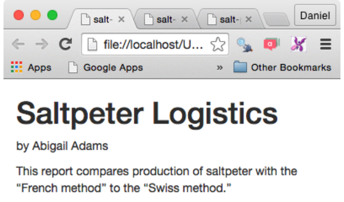
Step 1. Start with a short and simple document.

```
# Saltpeter Logistics

by Abigail Adams

This report compares
production of saltpeter
with the "French method"
to the "Swiss method."
```

Compile to HTML



Saltpeter Logistics

by Abigail Adams

This report compares production of saltpeter with the "French method" to the "Swiss method."

Step 2. Add the boilerplace chunk and a chunk to your start your wrangling. Test that it works. If not, go back and fix it.

~~~
# Saltpeter Logistics

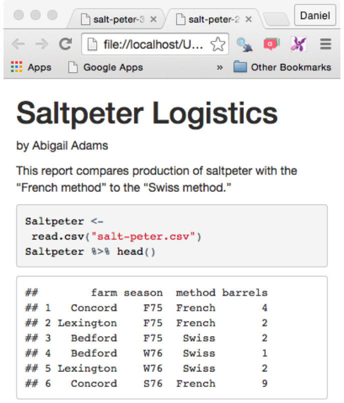
by Abigail Adams

```{r include=FALSE}
library(DataComputing)
~~~

This report compares
production of saltpeter
with the "French method"
to the "Swiss method."

```{r}
Saltpeter <-
 read.file("saltpeter.csv")
Saltpeter %>% head()
```
```

Compile to HTML


salt-peter-3
salt-peter-2
Daniel
file://localhost/U...
Apps
Google Apps
»
Other Bookmarks

Saltpeter Logistics

by Abigail Adams

This report compares production of saltpeter with the "French method" to the "Swiss method."

```
Saltpeter <-
 read.csv("salt-peter.csv")
Saltpeter %>% head()
```

```
##         farm season  method barrels
## 1    Concord    F75  French       4
## 2 Lexington    F75  French       2
## 3    Bedford    F75   Swiss       2
## 4    Bedford    W76   Swiss       1
## 5 Lexington    W76   Swiss       2
## 6    Concord    S76  French       9
```

Step 3. Add more to the wrangling chunk. Test that it works.

```
# Saltpeter Logistics

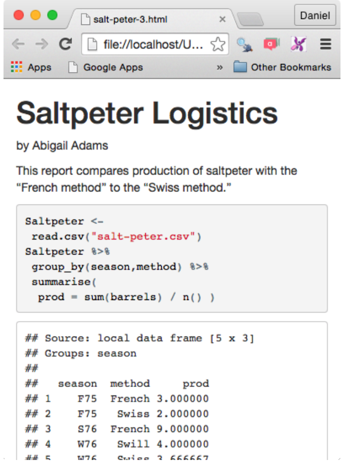
by Abigail Adams

```{r include=FALSE}
library(DataComputing)
```

This report compares
production of saltpeter
with the "French method"
to the "Swiss method."

```{r}
Saltpeter <-
 read.file("saltpeter.csv")
Saltpeter %>%
 group_by(season,method) %>%
 summarise(
  prod = sum(barrels) / n() )
```
```

Compile to HTML


salt-peter-3.html
Daniel
file://localhost/U...
Apps
Google Apps
»
Other Bookmarks

Saltpeter Logistics

by Abigail Adams

This report compares production of saltpeter with the "French method" to the "Swiss method."

```
Saltpeter <-
 read.csv("salt-peter.csv")
Saltpeter %>%
 group_by(season,method) %>%
 summarise(
  prod = sum(barrels) / n() )
```

```
## Source: local data frame [5 x 3]
## Groups: season
##
##   season  method     prod
## 1    F75  French 3.000000
## 2    F75   Swiss 2.000000
## 3    S76  French 9.000000
## 4    W76   Swill 4.000000
## 5    W76   Swiss 3.666667
```

Figure 4.5: Three steps in the write/compile cycle. At each step, the `.Rmd` file (shown in `monospace font`) is compiled into the `.html` format shown underneath.
```

DISTRIBUTING THE .RMD FILE IS HELPFUL when communicating with a technically savvy audience. A nice strategy for getting the benefits of both the easily-readable `.html` format and the original `.Rmd` file is to include the `.Rmd` file *inside* the `.html`, much as you might attach a file to an email message. The `DataComputing` package provides a way to do this easily by including the following chunk in the `.Rmd` file.

The document template `Data Computing simple` includes the source document by default.

~~~
```{r echo=FALSE, results="asis"}
cat("Source Documents: ")
DataComputing::includeSourceDocuments()
```
~~~

This chunk will embed the source file into your `.html`. The originating `.Rmd` file can be extracted by clicking on a link that is contained within the `.html` file.

4.1 *Exercises*

Problem 4.1
Markdown provides a simple way to produce section headers and sub-headers, italic and bold text, `monospaced` fonts suitable for computer commands, and even web links. A reference is available in RStudio at the menu HELP/MARKDOWN QUICK REFERENCE.

For each of the following, say how it will be rendered when the Markdown is rendered to HTML. (Hint: You can figure it out by reading the documentation, or you can put the text into an Rmd document and compile it!)

~~~~
*one*

**two**

* three

# Four

`five`

## Six

[seven](http://tiny.cc/dcf/index.html)
~~~~

Problem 4.2
What's wrong with the markup for each of these five chunks:

~~~~
    (a)        (b)        (c)

    '''{r}     """(r)     ```{r}
    9+7        9+7        9+7
    '''        """        ``

          (d)             (e)

.         ```{r} 9+7```   ```{r}
                          9+7
                          ````
~~~~

Problem 4.3
Treat the following lines as an `.Rmd` file which will be compiled to HTML.

~~~~
### An Introduction

Arithmetic is *easy*! For instance
```{r}
3 + 2
```
~~~~

Using paper and pencil, sketch out what the HTML document will look like when viewed in a web browser.

Problem 4.4
Here is a short list of names:

1. `DataComputing.org`
2. `ahab/whale.Rmd`
3. `ptth://world-bank.org`
4. `http://world-bank.org`
5. `//world-bank.org/index.html`
6. `world-bank.org/index.html`

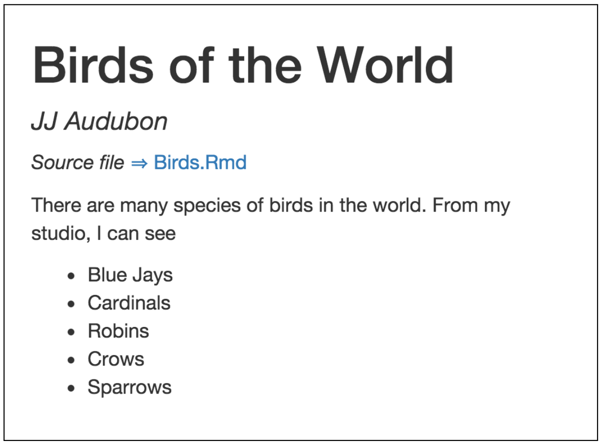
For each, say whether the name is in the allowed form for a possible URL, a possible file, neither, or both.

Problem 4.5
From the RStudio console, load the `DataComputing` package like this:

```
library(DataComputing)
```

Once this is done,

1. Open a new file using the File/New File/R Markdown ... menu item. Select "From Template" and then choose the *DataComputing simple* template.
2. Save the text that appears in the editor tab in a file named `Birds.Rmd`
3. Compile the `Birds.Rmd` file to HTML to verify that the template is working.
4. Edit the `Birds.Rmd` file to include contents that will make the compiled HTML file appear like this:



Birds of the World

JJ Audubon

Source file ⇒ Birds.Rmd

There are many species of birds in the world. From my studio, I can see

- Blue Jays
- Cardinals
- Robins
- Crows
- Sparrows