```
# Video and sample code for webscraping

library(tidyverse)
library(rvest)

# Import the web page
url <- "https://en.wikipedia.org/wiki/Murder_in_the_United_States_by_state"
h <- read_html(url)

# The class of the object is XML document
class(h)

# Here we know that the information is stored in an HTML table.
# You can see this in a line of code of the HTML document we showed earlier.
# <table class="wikitable sortable">

# The rvest package includes functions to extract nodes from HTML documents.
# The function html_nodes, plural, extracts all nodes of that type.
# And html_node extracts just the first node of that type.
# This gives us just the html code for that table
tab <- h %>% html_nodes("table")
tab <- tab[[2]]
tab

# In the code we just showed, you can definitely see a pattern,
# and writing code to extract just the data is very doable.
# In fact rvest includes a function precisely for this--
# for converting HTML tables into data frames.
tab <- tab %>% html_table
class(tab)

# We are now much closer to having a usable data table.
# Let's change the names of the columns, which are a little bit long,
# and then take a look. / You can see that we already
# have a data frame very close to what we want.
tab <- tab %>% setNames(c("state", "population", "total", "murders", "gun_ownership", "total_rate", "murder_rate", 
"gun_murder_rate"))
head(tab)

# However, we still have some data wrangling to do.
# For example, notice that some of the columns that are supposed to be numbers
# are actually characters.
# And what makes it even worse is that some of them have commas,
# so it makes it harder to convert to numbers.
# Before we continue with this, we're going to learn a little bit more
# about general approaches to extracting information from websites.
```