Day 05: Scraping Part 2

Erin Rossiter

20 February, 2023

Announcements

Announcements

- PS03 graded

Announcements

- PS03 graded
  - please review comments, especially on final project

Announcements

- PS03 graded
  - please review comments, especially on final project
- Final projects

Announcements

- PS03 graded
  - please review comments, especially on final project
- Final projects
  - get started!

Announcements

- PS03 graded
  - please review comments, especially on final project
- Final projects
  - get started!
  - if your answers to PS03 were brief/vague, I strongly recommend you get started with those initial steps this week

Announcements

- PS03 graded
  - please review comments, especially on final project
- Final projects
  - get started!
  - if your answers to PS03 were brief/vague, I strongly recommend you get started with those initial steps this week
  - I won't formally check in again until 3/28

Announcements

- PS03 graded
  - please review comments, especially on final project
- Final projects
  - get started!
  - if your answers to PS03 were brief/vague, I strongly recommend you get started with those initial steps this week
  - I won't formally check in again until 3/28
    - project management practice

Announcements

- PS03 graded
  - please review comments, especially on final project
- Final projects
  - get started!
  - if your answers to PS03 were brief/vague, I strongly recommend you get started with those initial steps this week
  - I won't formally check in again until 3/28
    - project management practice
- PS04 due tomorrow

Today's plan

Today's plan

- Last time

Today's plan

- Last time
  - basics of websites/html

Today's plan

- Last time
  - basics of websites/html
  - rvest

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions
    - Condition handling (or you might see it called exception handling)

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions
    - Condition handling (or you might see it called exception handling)
    - (Conceptual stuff, so more slides)

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions
    - Condition handling (or you might see it called exception handling)
    - (Conceptual stuff, so more slides)
  - Extracting data from more complex websites and JSON

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions
    - Condition handling (or you might see it called exception handling)
    - (Conceptual stuff, so more slides)
  - Extracting data from more complex websites and JSON
  - Selenium

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions
    - Condition handling (or you might see it called exception handling)
    - (Conceptual stuff, so more slides)
  - Extracting data from more complex websites and JSON
  - Selenium
- Next time

Today's plan

- Last time
  - basics of websites/html
  - rvest
  - more practice with the basics
    - Functionalizing
    - Data management
    - Text parsing
- Today
  - We're also using this unit to cover additional R skills that are particularly relevant
    - Conditions
    - Condition handling (or you might see it called exception handling)
    - (Conceptual stuff, so more slides)
  - Extracting data from more complex websites and JSON
  - Selenium
- Next time
  - APIs

Condition handling

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2...

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...
- To handle this, we need to:

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...
- To handle this, we need to:
  - Understand errors, warnings, etc. that might occur

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...
- To handle this, we need to:
  - Understand errors, warnings, etc. that might occur
  - Understand how to handle these problems

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...
- To handle this, we need to:
  - Understand errors, warnings, etc. that might occur
  - Understand how to handle these problems
    - Tools in R

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...
- To handle this, we need to:
  - Understand errors, warnings, etc. that might occur
  - Understand how to handle these problems
    - Tools in R
    - Defensive strategies

Motivation

- Imagine you're scraping 10,000 websites, and your computer starts the task at night, you wake up in the morning and the code threw an error on iteration 2... that sucks
  - Same story beyond scraping
  - Cleaning 10 million observations of a variable with a for loop...
  - Simulating a model 10K times...
- To handle this, we need to:
  - Understand errors, warnings, etc. that might occur
  - Understand how to handle these problems
    - Tools in R
    - Defensive strategies
  - Maybe we even implement errors/warnings outselves

How does R communicate with you? Three "conditions"

1. Errors

- Force all execution to terminate immediately

1. Errors

- Force all execution to terminate immediately

```
my_add <- function(x, y){
  print("We started the function....")
  my_sum <- x + y
  print("...we ended the function")
  return(my_sum)
}
my_add(x = 1, y = "B")
```

```
## [1] "We started the function...."
```

```
## Error in x + y: non-numeric argument to binary operator
```

Errors

- You can communicate errors too! To yourself or others using your code

Errors

- You can communicate errors too! To yourself or others using your code
  - `stop()` function for custom errors

Errors

- You can communicate errors too! To yourself or others using your code
  - `stop()` function for custom errors
  - note this is a redundant example because R already has an error for this :)

```
my_add <- function(x, y){
  if(!is.numeric(x) | !is.numeric(y)){
    stop("both inputs need to be numeric")
  }
  return(x + y)
}
my_add(x = 1, y = "B")
```

```
## Error in my_add(x = 1, y = "B"): both inputs need to be numer
```

Example

- Another example, R would otherwise try to execute this!

Example

- Another example, R would otherwise try to execute this!

```
my_add_vectors <- function(x, y){
  if(length(x) != length(y)){
    stop("both inputs need to be the same length")
  }
  return(x + y)
}
my_add_vectors(x = 1, y = 1:10)
```

```
## Error in my_add_vectors(x = 1, y = 1:10): both inputs need to
```

2. Warnings

- Warnings communicate *potential* problems

2. Warnings

- Warnings communicate *potential* problems
- Unlike errors, code continues

2. Warnings

- Warnings communicate *potential* problems
- Unlike errors, code continues

```
str_x <- c("1", "72", "300", "hi", "1")
num_x <- as.numeric(str_x)
```

```
## Warning: NAs introduced by coercion
```

```
num_x
```

```
## [1]   1  72 300  NA   1
```

Warnings

- Like errors, you can implement your own warnings

Warnings

- Like errors, you can implement your own warnings
  - `warning()` function

Warnings

- Like errors, you can implement your own warnings
  - `warning()` function

```
my_add_vectors <- function(x, y){
  if(length(x) != length(y)){
    warning("Output will use vector recyling")
  }
  return(x + y)
}
my_add_vectors(x = 1, y = 1:10)
```

```
## Warning in my_add_vectors(x = 1, y = 1:10): Output will use v
```

```
##  [1]  2  3  4  5  6  7  8  9 10 11
```

3. Messages

- You might see messages giving an informative output

3. Messages

- You might see messages giving an informative output
  - Like warnings, code will execute

3. Messages

- You might see messages giving an informative output
  - Like warnings, code will execute
  - Unlike warnings, messages are simply informative

3. Messages

- You might see messages giving an informative output
  - Like warnings, code will execute
  - Unlike warnings, messages are simply informative
- Example:/pause
  - I didn't specify the variable on which to join, so the function informed me what it deduced should be the joining variable

3. Messages

- You might see messages giving an informative output
  - Like warnings, code will execute
  - Unlike warnings, messages are simply informative
- Example:/pause
  - I didn't specify the variable on which to join, so the function informed me what it deduced should be the joining variable

```
# simulate data
df1 <- data.frame(id = 1:10, age = sample(20:80, 10))
df2 <- data.frame(id = 1:10, inc = sample(20:200, 10))
df_combined <- plyr::join(x = df1, y = df2)
```

```
## Joining by: id
```

Example

```
library(quanteda)
```

```
## Package version: 3.2.3
## Unicode version: 14.0
## ICU version: 70.1
```

```
## Parallel computing: 8 of 8 threads used.
```

```
## See https://quanteda.io for tutorials and examples.
```

```
library(MCMCpack)
```

```
## Loading required package: coda
```

```
## Loading required package: MASS
```

```
## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)
```

```
## ## Copyright (C) 2003-2023 Andrew D. Martin, Kevin M. Quinn,
```

```
## ##
## ## Support provided by the U.S. National Science Foundation
```

```
## ## (Grants SES-0350646 and SES-0350613)
```

Messages

- Again, you can use messages in your code

Messages

- Again, you can use messages in your code
- Why not just use `print`?

Messages

- Again, you can use messages in your code
- Why not just use `print`?
  - `message()` provides an error state that can be used with `tryCatch()...` our next topic

Messages

- Again, you can use messages in your code
- Why not just use `print`?
  - `message()` provides an error state that can be used with `tryCatch()...` our next topic

```
message("Here's a message")
```

```
## Here's a message
```

```
print("Here's printed text")
```

```
## [1] "Here's printed text"
```

How can you work with these “conditions”?

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!
- We can be prepared for this

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!
- We can be prepared for this
  - `try()`

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!
- We can be prepared for this
  - `try()`
    - continue to execute the code even when an error occurs

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!
- We can be prepared for this
  - `try()`
    - continue to execute the code even when an error occurs
  - `tryCatch()`

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!
- We can be prepared for this
  - `try()`
    - continue to execute the code even when an error occurs
  - `tryCatch()`
    - specify a "handler function" to handle the problem a certain way when a condition occurs

Condition handling

- Recall, sometimes R throws errors, but we don't want our code to stop!
- We can be prepared for this
  - `try()`
    - continue to execute the code even when an error occurs
  - `tryCatch()`
    - specify a "handler function" to handle the problem a certain way when a condition occurs

Without `try()`

- Allows the code to continue even if there's an error

Without `try()`

- Allows the code to continue even if there's an error
- (More examples soon)

Without `try()`

- Allows the code to continue even if there's an error
- (More examples soon)

```
f_notry <- function(){
  # do some stuff that might throw an error
  2 + "B"
  # return function output
  return("example output")
}
f_notry()
```

```
## Error in 2 + "B": non-numeric argument to binary operator
```

With `try()`

```
f_withtry <- function(){
  # do some stuff that might throw an error
  try(2 + "B")
  # return function output
  return("example output")
}
f_withtry()
```

```
## Error in 2 + "B" : non-numeric argument to binary operator
```

```
## [1] "example output"
```

`tryCatch()`

- Allows the code to continue even if there's an error **and** "handles" the error too

`tryCatch()`

- Allows the code to continue even if there's an error **and** "handles" the error too
- (More examples soon)

tryCatch()

- Allows the code to continue even if there's an error **and** "handles" the error too
- (More examples soon)

```
f_withtrycatch <- function(){
  # do some stuff that might throw an error
  tryCatch(x <- 2 + "8",
    error = function(e){
      print(e)
    },
    x <- 2 + 8)
  # return function output
  return(x)
}
out <- f_withtrycatch()
```

```
## <simpleError in 2 + "8": non-numeric argument to binary opera
```

```
out
```

```
## [1] 10
```

Script day05-conditionhandling.R

Extracting data from more complex websites and JSON

Programs interfacing with programs

https://www.cnn.com/election/2018/results

Programs interfacing with programs

https://www.cnn.com/election/2018/results

- Let's look at this with "Inspect element"

Programs interfacing with programs

https://www.cnn.com/election/2018/results

- Let's look at this with "Inspect element"
- Not so easy!

Programs interfacing with programs

https://www.cnn.com/election/2018/results

- Let's look at this with "Inspect element"
- Not so easy!
- But this data has to be coming from somewhere...

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts
- The network tab is there to help diagnose all of the different connections as websites operate

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts
- The network tab is there to help diagnose all of the different connections as websites operate
  - Think of it as a “log” of activity.

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts
- The network tab is there to help diagnose all of the different connections as websites operate
  - Think of it as a "log" of activity.
- For some websites, we can find where data itself is coming from.

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts
- The network tab is there to help diagnose all of the different connections as websites operate
  - Think of it as a "log" of activity.
- For some websites, we can find where data itself is coming from.
  - Let's take a look at the network button as this website is loaded.

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts
- The network tab is there to help diagnose all of the different connections as websites operate
  - Think of it as a "log" of activity.
- For some websites, we can find where data itself is coming from.
  - Let's take a look at the network button as this website is loaded.
  - I found this link:

Network tab for Chrome

- As pages are rendered, they often call to other servers and execute scripts
- The network tab is there to help diagnose all of the different connections as websites operate
  - Think of it as a "log" of activity.
- For some websites, we can find where data itself is coming from.
  - Let's take a look at the network button as this website is loaded.
  - I found this link:
    - https://data.cnn.com/ELECTION/2018November6/IN/county/S.json
  - Whoa look at that organized data!

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.
- JSON (JavaScript Object Notation) is a generic format for any data object being passed around.

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.
- JSON (JavaScript Object Notation) is a generic format for any data object being passed around.
- It is very similar to **lists** in R.

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.
- JSON (JavaScript Object Notation) is a generic format for any data object being passed around.
- It is very similar to **lists** in R.
  - **key**: name of element

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.
- JSON (JavaScript Object Notation) is a generic format for any data object being passed around.
- It is very similar to **lists** in R.
  - **key**: name of element
  - **value**: element contents

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.
- JSON (JavaScript Object Notation) is a generic format for any data object being passed around.
- It is very similar to **lists** in R.
  - **key**: name of element
  - **value**: element contents
- Often called "key-value" pair

JSON

- Unlike the presidential library and ND website, lot of the more complex data-driven websites you want these days are running JavaScript.
- JSON (JavaScript Object Notation) is a generic format for any data object being passed around.
- It is very similar to **lists** in R.
  - **key**: name of element
  - **value**: element contents
- Often called "key-value" pair

```
'{"name":"John", "age":30, "car":null}'
```

JSON Example 2

Can nest like lists in R, too!

JSON Example 2

Can nest like lists in R, too!

- `students` is a list, each element itself has multiple elements

JSON Example 2

Can nest like lists in R, too!

- `students` is a list, each element itself has multiple elements

```
'{
  "class': "Programming",
  "num_students" : 15,
  "year": 2023,
  "students": [
    {
      "name": "Adriana",
      "program": "PoliSci",
      "fields": ["Comparative", "Methodology"]
    },
    {
      "name": "Emily",
      "program": "PoliSci",
      "fields": ["American", "Methodology"]
    },
    {
      "name": "Abigail",
      "program": "PoliSci",
      "fields": ["American", "Methodology"]
    }
  ]
}'
```

Script day05-JSON.R

Selenium

Selenium

- The website you want may...

Selenium

- The website you want may...
  - have forms

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques
  - involve a giant backend database you don't/can't get all of

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques
  - involve a giant backend database you don't/can't get all of
- Selenium is software designed to allow you to operate a browser like you are a person

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques
  - involve a giant backend database you don't/can't get all of
- Selenium is software designed to allow you to operate a browser like you are a person
- Anything you can access/get via normal browsing behavior is accessible

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques
  - involve a giant backend database you don't/can't get all of
- Selenium is software designed to allow you to operate a browser like you are a person
- Anything you can access/get via normal browsing behavior is accessible
- I am not going to help you get configured right now, but here is a place to start

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques
  - involve a giant backend database you don't/can't get all of
- Selenium is software designed to allow you to operate a browser like you are a person
- Anything you can access/get via normal browsing behavior is accessible
- I am not going to help you get configured right now, but here is a place to start
  - [https://cran.r-project.org/web/packages/RSelenium/vignettes/basics.html]
  - Script will be demonstration only

Selenium

- The website you want may...
  - have forms
  - be intentionally unfriendly to previous techniques
  - involve a giant backend database you don't/can't get all of
- Selenium is software designed to allow you to operate a browser like you are a person
- Anything you can access/get via normal browsing behavior is accessible
- I am not going to help you get configured right now, but here is a place to start
  - [https://cran.r-project.org/web/packages/RSelenium/vignettes/basics.html]
  - Script will be demonstration only
  - Seriously, it took me 45mins to get my machine set up, please do not try right now, just enjoy the demo :)