Scraping Nuclear Reactors

Data Computing

Computing Project

In this project, you're going to scrape data about nuclear reactors in various courntries from Wikipedia.

Tables in HTML pages

Go to the page `http:://en.wikipedia.org/wiki/List_of_nuclear_reactors`. Find the reactor list for Japan.

Although such lists are in a visual tabular format, they do not have a simple data-table structure. The tables are organized using HTML tags, which provide much more flexibility for visual appearance.

```
<table class="wikitable sortable">
<tr>
<th rowspan="2" style="background:#FFDEAD;">Name</th>
... and so on ...
</tr>
<tr>
<td>Fukushima Daiichi</td>
<td>1</td>
... and so on ...
</tr>
```

Compare the human-readable version of the table with the HTML markup. You'll see that the data is there, but there is a lot of extraneous material and the arrangement is set not by position in a spreadsheet layout but by *HTML tags*[1] like `<td>` and `<tr>`.

Parsing HTML into a data table

```r
library(rvest)
library(lubridate)
page <- "http://en.wikipedia.org/wiki/List_of_nuclear_reactors"
table_nodes <- page %>%
  read_html() %>%
  html_nodes("table")
table_list <-
  html_table(table_nodes[1:30], fill = TRUE)
```

The `table_list` object is not quite a data table; it is a *list* of data tables. Here are some of the operations you can apply to lists:

Description	Syntax	Example
How many elements in the list	`length(`*table*`)`	`length(table_list)`
Grab a single element	*table*`[[`*element number*`]]`	`table_list[[20]]`

[1] A markup indicator, analogous to ∗ or ### or `[text](line)` in markdown.

1) Find the table element

Start with `head(table_list[[5]])` and go down the list until you find the table for Japan. Keep in mind that the tables are listed by number in the same order that they appear on the page. As of the time of this writing,[2] `table_list[[5]]` is for Austria, so you'll have to go a good distance down the table to get to Japan.

```
table = table_list[[21]]  # change index for Japan
names(table)
```

2) Look at it using `View()`

The contents of row 1 don't refer to a case but to the variable names. To clean this table, you will want to create meaningful variable names and then delete row 1. You may need to refer to the original HTML document to figure out what are appropriate names.

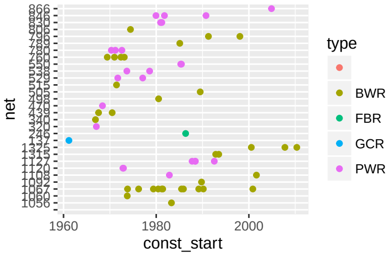
Here are some examples of the types of statements you might find helpful for fixing the variable names.

```
new_names <- c("first", "second", "third")
names(table) <- new_names # reset the variable names
table <- table[-1, ] # drop the first row
```

A quick visualization

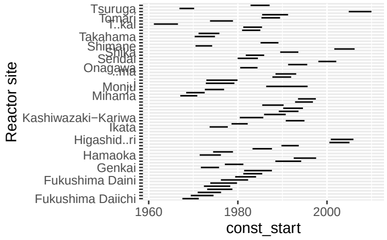
Plot out electrical capacity versus date of commissioning. Remember to turn the commissioning date into a genuine *date object*[3]. Color the points by the *type* of reactor, e.g., BWR, PWR, or FBR.[4]

Interpretation: the net capacity of nuclear power plants in Japan tended to increase over time (but then plateaued in recent years).



Construction delays

Make an informative graphic that shows how long it took between start of construction and commissioning for each nuclear reactor in Japan (or another country of your choice). One possibility: use reactor name vs date as the frame. For each reactor, set the glyph to be a line extending from start of construction to commissioning. You can do this with `geom_segment()` using name as the y coordinate and time as the x coordinate.



[2] Wikipedia articles are works in progress. Over a period of even a few days they may have been modified substantially.

[3] A type of R object representing points in time but allowing plotting, extraction of components, and mathematical operations to be carried out.

[4] Boiling water reactor, pressurized water reactor, fast breeder reactor, respectively