# Homework 3: Super Scalper Scrape

*36-350, Fall 2014*

***General instructions:*** Your homework must be submitted in R Markdown format. We will not (indeed, cannot) grade homeworks in other formats. Your responses must be supported by both textual explanations and the code you generate to produce your result. (Just examining your various objects in the "Environment" section of RStudio is insufficient – you must use scripted commands.) Do *not* include the text of the questions themselves in the write-up you submit.

**Background**: In lab 3, you examined the website for the 100 Richest People in the world according to *Forbes* magazine. For this homework, we'll show you how you might become rich yourself by collecting the schedule for the upcoming NHL season, including the links to Ticketmaster, so that you can corner the resale market.

Just kidding! There's no way with what we've taught you so far that you'll be able to get past the anti-bot measures on Ticketmaster without defeating the CAPTCHA systems that were designed and built right here at CMU. What you can do, though, is reassemble this series of games in an R data frame for machine-readable use. A To do so, you will use regular expressions to extract the useful information from the surrounding HTML code.

1. Use the `readLines` command to load the file `NHLHockeySchedule2.html` into a character vector called `nhl1415`.

    a. How many lines does it contain?
    b. What is the total number of characters in the file?
    c. What is the maximum number of characters in a line?

2. Open `NHLHockeySchedule2.html` as a web page. You should see the game table on the screen. There are 1230 regular-season games scheduled. Who is playing in the first game? In the final game?

3. Now, open `NHLHockeySchedule2.html` in a text editor. What line in the file corresponds to game 1? Which line corresponds to game 1230? How do each of these lines begin?

4. Our goal is to extract the date, game time (in Eastern Time), away and home teams, and URL for purchasing tickets. Before continuing with this task, do you notice anything about the ticket link URLs compared to the info for the rest of the game?

5. Write a regular expression to capture the date. Use `grep()` to check that this has exactly 1230 matches and that the first and last locations match the first and last games.

6. Using the regular expression above, and the functions `regexp` and `regmatches`, extract all the dates from the text and create a corresponding vector `date`. Save this for a further step.

7. Now, identify the away and home teams with regular expressions for each. Use the HTML around these names to guide your search. You may have to add escape marks to properly recognize some characters – for example, periods must be escaped as `\\.` in your expression. Extract and save these values in their own vectors.

8. Identify the game time in the code and create a regular expression for it. Note that there are two times – the local time and the Eastern time. Make sure your expression gets the Eastern time using clues from the HTML. Save it.

9. Finally, identify the URL that corresponds to the "TICKETS>" button. Create a regular expression for this and search for it in the code. Make sure that you have exactly 1230 of them! Check that the team specified in the link has the home team in it (for the most part).

10. Construct a data frame on these five variables. Print the frame from rows 1221 to 1230. Does the data match that in the last 10 rows of the table as seen from your web browser?