Parsing XML data

Unit 3 - Lab 6

Directions: Follow along with the slides and answer the questions in **BOLDED** font in your journal.

# Working with XML

* In the previous lab, you used the XML package to scrape some data from a web table.
* Now we'll use the XML package to *parse* some actual XML data.
* XML has a very different structure than data we're used to.
* This is helpful so that websites can have greater control over where different pieces of data are placed on the page.
* It also requires us to scrape the data slightly differently than before.
* So load up the XML package again and let's get scrapin'!

library(XML)

# XML Lingo

* Scraping XML involves a slightly different lingo.
* *metadata* is information about our actual data.
* *tags* are how place values into our data's hierarchy.
* *nodes* are tags that contain other tags.
* *parse* just means navigate through the tags in the data.
* *children* are sets of nodes that are contained within tags.
* Try not to let the syntax scare you.

# Just like before:

-Type (Don't copy/paste) the following [url to our data](http://web.ohmage.org/mobilize/resources/ids/data/mountains.xml)

data\_url <- "http://web.ohmage.org/mobilize/  
 resources/ids/data/mountains.xml"

* But notice this time, our readHTMLTable() function doesn't work

readHTMLTable(data\_url)

* That's because we need to *parse* or navigate through the data first.

xmlParse(data\_url)

# Save the Children!

* Save the parsed XML as xml\_file

xml\_file <- xmlParse(data\_url)

* Notice that the first (and last) tags are mountainpeaks.
* Imagine that these tags are a bag that hold all of the rest of the *nodes*. We call this set of *nodes* ''*children*''.
* We want to save these *children* so we can use their information.

xml\_children <- xmlChildren(xml\_file)

# Finding our data

* You might remember that XML can contain *metadata*, which is information about the data. Since we want to eventually analyze this data with R, we need to find a way to leave the *metadata* and take only the actual data.
* Start by looking at the last end tag in the xml\_children:

xml\_children

* Notice that the last tag is for mountainpeaks and the next to last is our data.

# Getting our data

* We can take everything within the mountainpeaks tag by running:

xml\_children[["mountainpeaks"]]

* To get into the data tag, we can run something simliar:

xml\_children[["mountainpeaks"]][["data"]]

* We're getting close! We really just want to go one more step to the mountains node.
* **Write the code you would use, that is similar to the above code, to look at the info contained in the mountains node.**

# We've arrived at data.

* We've finally arrived at our actual data

xml\_children[["mountainpeaks"]][["data"]][["mountains"]]

* To convert and save our data as our familiar R data frame, run:

mountains\_xml <- xml\_children[["mountainpeaks"]][["data"]][["mountains"]]

xmlToDataFrame(mountains\_xml)

* Assign this data the name mountains.

# A deeper look

* Just for kicks run the following code.

xml\_children[["mountainpeaks"]][["data"]][["mountains"]][["mountain"]]

* **What information is displayed when you run this code?**
* **Now View your mountains data and write down which row matches the above code's information.**

# Clean it up!

* Just like when we scraped the HTML table, we need to clean up our data before we can use it.
* **Check the names of the mountains data to make sure they look correct.**
* **Check the structure (str()) of the data. If it's incorrect, run:**

var\_types <- c("character","character",  
 "character", "numeric",   
 "numeric", "numeric", "numeric",   
 "numeric", "numeric", "numeric")

mountains <- xmlToDataFrame(mountains\_xml,  
 colClasses=var\_types)

# When you're done.

* **After cleaning your data, run the following command and write down the output:**

favstats(~elev\_ft, data=mountains)

* If you'd like, you can save your data by running:

save(mountains, file='mountains.rda')