Setup Recommendation for Node.js and XPath

Although the projects have you run code in your browser, it is recommended to have Node.js install on your system to run the code quality checker JSHint. If you want to install Node.js and the npm package manager for your group project later, follow the installation instructions here.

http://web.stanford.edu/class/cs142/install.html

Optional JSHint installation for code quality checker for your projects:

Once you have Node.js installed, and if you want to run JSHint for your extra credit project 3 for example, create a directory `project3` and extract the contents of the zip file on the class webpage into the directory.

You can fetch the code quality tool, JSHint, by running the following command in the `project3` directory:

```
npm install
```

This will fetch JSHint into the `node_modules` subdirectory.

You will be able to run it on all the JavaScript files in `project3` directory by running the command:

```
npm run jshint
```

The code you submit should start with `"use strict";` and run JSHint without warnings.

References:

http://web.stanford.edu/class/cs142/install.html

https://nodejs.org/en/download/

https://www.mongodb.com/download-center#production

Additional Links on the Class Webpage:

Node JS with MongoDB/PostgreSQL/SQL Server:

Node JS:

https://nodejs.org/en/about/

https://nodejs.org/en/download/

https://nodejs.org/en/download/package-manager/

MongoDB:

https://docs.mongodb.com/?_ga=1.237374408.990523453.1486931228

MongoDB Download:

https://www.mongodb.com/download-center#production

Set up Guide:

Node JS with Mongo DB Setup Guide:

http://web.stanford.edu/class/cs142/install.html

Node JS with PostgreSQL Setup Guide:

http://eecs.csuohio.edu/~sschung/cis612/NodeAngularSetUpGuide.pdf

Sample Web Application Using Node JS with MS SQL Server and Mongo DB:

http://eecs.csuohio.edu/~sschung/cis612/WebApplicationExampleNodeJSSQLServerMongoDB.pdf

Xpath Set Up

https://www.npmjs.com/package/xpath

DOM 3 Xpath implemention and helper for node.js.

xpath

DOM 3 XPath 1.0 implementation and helper for JavaScript, with node.js support.

Install

Install with npm:

```
npm install xpath
```

xpath is xml engine agnostic but I recommend to use xmldom:

```
npm install xmldom
```

Your first xpath:

```
var xpath = require('xpath')
  , dom = require('xmldom').DOMParser
```

```
var xml = "<book><title>Harry Potter</title></book>"
var doc = new dom().parseFromString(xml)
var nodes = xpath.select("//title", doc)

console.log(nodes[0].localName + ": " + nodes[0].firstChild.data)
console.log("node: " + nodes[0].toString())
```

➞

```
title: Harry Potter
Node: <title>Harry Potter</title>
```

Get text values directly

```
var xml = "<book><title>Harry Potter</title></book>"
var doc = new dom().parseFromString(xml)
var title = xpath.select("//title/text()", doc).toString()

console.log(title)
```

➞

```
Harry Potter
```

Namespaces

```
var xml = "<book><title xmlns='myns'>Harry Potter</title></book>"
var doc = new dom().parseFromString(xml)
var node = xpath.select("//*[local-name(.)='title' and namespace-uri(.)='myns']", doc
)[0]

console.log(node.namespaceURI)
```

➞

```
myns
```

Namespaces with easy mappings

```
var xml = "<book xmlns:bookml='http://example.com/book'><bookml:title>Harry Potter</b
ookml:title></book>"
```

```
var select = xpath.useNamespaces({"bookml": "http://example.com/book"});


console.log(select('//bookml:title/text()', doc)[0].nodeValue);
```

➞

```
Harry Potter
```

Default namespace with mapping

```
var xml = "<book xmlns='http://example.com/book'><title>Harry Potter</title></book>"
var select = xpath.useNamespaces({"bookml": "http://example.com/book"});


console.log(select('//bookml:title/text()', doc)[0].nodeValue);
```

➞

```
Harry Potter
```

Attributes

```
var xml = "<book author='J. K. Rowling'><title>Harry Potter</title></book>"
var doc = new dom().parseFromString(xml)
var author = xpath.select1("/book/@author", doc).value


console.log(author)
```

➞

```
J. K. Rowling
```

References:

https://www.npmjs.com/package/xpath

https://github.com/goto100/xpath

https://github.com/yaronn/xpath.js/issues

Alternatively,

XPath is already built in any recent Web Browser. You can use it in Java Script in your html document. See the MDN link in XPath section in the Lecture Notes Section for the detail examples.

You can use `document.evaluate`:

Evaluates an XPath expression string and returns a result of the specified type if possible.

It is w3-standardized in and documented at:

http://www.w3.org/TR/DOM-Level-3-XPath/xpath.html

https://developer.mozilla.org/en-US/docs/Web/API/Document.evaluate

example:

```
function getElementByXpath(path) {

  return document.evaluate(path, document, null, XPathResult.FIRST_ORDERED_NODE_
TYPE, null).singleNodeValue;

}


console.log( getElementByXpath("//html[1]/body[1]/div[1]") );

<div>foo</div>
```

There's also a great introduction on mozilla developer network (MDN) (also on class webpage)

https://developer.mozilla.org/en-US/docs/Introduction_to_using_XPath_in_JavaScript#document.evaluate

XML Parser in Java Script

To parse (read and convert) the XML found through using GeoNames' FindNearestAddress.

If your XML is in a string variable called `txt` and looks like this:

```
<address>

 <street>Roble Ave</street>

  <mtfcc>S1400</mtfcc>

  <streetNumber>649</streetNumber>

  <lat>37.45127</lat>

  <lng>-122.18032</lng>
```

```
  <distance>0.04</distance>
  <postalcode>94025</postalcode>
  <placename>Menlo Park</placename>
  <adminCode2>081</adminCode2>
  <adminName2>San Mateo</adminName2>
  <adminCode1>CA</adminCode1>
  <adminName1>California</adminName1>
  <countryCode>US</countryCode>
 </address>
```

Then you can parse the XML with Javascript DOM like this:

```
if (window.DOMParser)
  {
    parser = new DOMParser();
    xmlDoc = parser.parseFromString(txt, "text/xml");
  }
else // Internet Explorer
  {
    xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
    xmlDoc.async = false;
    xmlDoc.loadXML(txt);
  }
```

And get specific values from the nodes like this:

```
//Gets house address number
xmlDoc.getElementsByTagName("streetNumber")[0].childNodes[0].nodeValue;

//Gets Street name
xmlDoc.getElementsByTagName("street")[0].childNodes[0].nodeValue;

//Gets Postal Code
xmlDoc.getElementsByTagName("postalcode")[0].childNodes[0].nodeValue;
```

DOM and XPath for Python

Setting Up DOM with XPath for Python:

http://docs.python-guide.org/en/latest/scenarios/scrape/

BeautifulSoup Documentation for HTML, XML for Python:

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Python XPath Guide:
https://github.com/helloitsim/InstAnalytics/blob/fdec9d27896f4b6e61acd9dc56134ad556144057/InstAnalytics.py

Web scapping with DOM, XPath for Python with BeautifulSoup (From the Stanford Class)

http://web.stanford.edu/~zlotnick/TextAsData/Web_Scraping_with_Beautiful_Soup.html

Web Scrapping with XPath in Python

http://nbyim.com/monitor-instagram-accounts-without-using-api/

https://github.com/helloitsim/InstAnalytics/blob/fdec9d27896f4b6e61acd9dc56134ad556144057/README.md

https://github.com/helloitsim/InstAnalytics/blob/fdec9d27896f4b6e61acd9dc56134ad556144057/InstAnalytics.py