Who am I?

Lawrence Yao
UNSW casual staff
Developer Analyst at YTML Consulting

Email me if you need technical help or suggestions for your SENG project.

Continuing education

Employers will ask:
- What is exciting?

Your response:

When learning coding, be brave

Web Technology Overview
with a focus on JavaScript-based technologies

Lawrence Yao
l.yao@unsw.edu.au

Web Technology Architecture

From: http://www.w3.org/Consortium/techstack-desc.html

Web Browser Technology Architecture

From: https://hsivonen.fi/web-stack/
JavaScript

By: David Flanagan
Publisher: O'Reilly Media, Inc.
Pub. Date: May 3, 2011

• Highly recommended!!!
• Online access through UNSW library
• Learn all about the JavaScript programming language

JavaScript on client and server

Internet (the Cloud)

Server-side
JavaScript
Web server
with Node.js

Client-side JavaScript
within Chrome, Firefox, Internet Explorer...

Companion Web application

• Mash: The “supported” Web application
  - https://github.com/mlawry/unsw-cse-seng-mash-webapp
  - Covers all technologies discussed
  - Study it in your own time
  - Ask me any questions about it via email

Server-side JavaScript

http://nodejs.org/

• This simple web server written in Node responds with “Hello World” for every request.

```javascript
var http = require('http');
http.createServer(function(req, res) {
  res.writeHead(200, {
    'Content-Type': 'text/plain'
  });
  res.end('Hello World!');
}).listen(1337, '127.0.0.1');
console.log('Server running at http://127.0.0.1:1337/');
```

Node.js

• How does it work? Read these tutorials
  - http://www.nodebeginner.org/
    • Highly recommended, but not entirely free
    • Provides enough knowledge to understand MASH
    • Outdated (v0.4.5)
• Official API
  - http://nodejs.org/api/
**Classic LAMP stack**

- Apache Httpd (Web server)
- PHP (server-side programming)
- MySQL (database)
- Linux (operating system)

**From LAMP to Node.js**

- Apache Httpd (Web server)
- PHP (server-side programming)
- MySQL (database)
- Linux (operating system)

**From LAMP to Node.js**

- Apache Httpd (Web server)
- PHP (server-side programming)
- MySQL (database)
- Linux (operating system)

**Node.js architecture**

- JavaScript Engine (Google V8)
- Event Loop Library
- Thread Pool Library
- Stream Socket Library
- HTTP Parser
- DNS Resolver

**HTML DOM**

- To see the Document Object Model (DOM)
  - Open Web browser
  - Go to any Web page
  - Press F12 on keyboard to bring up the “Web Developer Tools” (works on Windows)
  - Type JavaScript directly in the Console
- DOM can be changed without reloading entire Web page
  - Manually: Web Developer Tools
  - Programmatically: JavaScript

**Client-side JavaScript**
Cascading Style Sheets

- HTML: content; CSS: formatting
- Colour, margin, position, animation, etc...
- Specified in-line or in a separate .css file
- CSS can be changed without reloading
  - Manually: Web Developer Tools
  - Programmatically: JavaScript

CSS selectors

- Tag: Select all DOM elements with <Tag>
  - "p", "h1"
- Tag#ID: Selects a unique <Tag> element with the given ID
  - "p#para", "p#para2"
- Tag.Class: Selects all <Tag> elements with the given Class
  - "h1.special", "p.special"
- #ID: Selects a unique element (any tag) with the given ID
  - "#para", "#para2"
- .Class: Selects all elements having the given Class
  - "special"

CSS style sheet (.css)

body { 
  line-height:1; 
  text-align:left; 
  margin-top:0 
}

menu.small,ol,ul { 
  list-style:none 
}

.jsyt-embed-thumbnail { 
  width:100%; 
  height:100%; 
  cursor:pointer; 
  background-size:cover; 
  background-repeat:no-repeat; 
  background-position:center 
}

JavaScript

- W3C DOM specification: http://www.w3.org/DOM/DOMTR

```javascript
1 < !DOCTYPE html>
2 < html>
3   < body>
4     < h1 > A Heading 1 </ h1 >
5     < p id=" para " > A paragraph of content </ p >
6   </ body>
7 </ html>
8
1 var domElem = document.getElementById("para");
2 domElem.setAttribute('style', 'color: red');
```

Web browser

- Provides an environment to run JavaScript
  - The JavaScript code changes the DOM/CSS
  - DOM/CSS changes >> dynamic Web page
- Ties things together
  - Reads HTML and CSS, creates DOM, applies CSS, executes JavaScript, interprets result as the final Web page
Ajax

- Asynchronous JavaScript and XML
- Partially update a Web page in reaction to some event
  - No need to refresh entire Web page
  - Smoother user experience (less disruptive)
- Relies on Web browser functionality
  - Exposed as XMLHttpRequest (XHR) API
- Usually send/receive JSON in background
  - JSON data format: [http://json.org/](http://json.org/)

Front-End Frameworks

- A compilation of complementary display elements that works like a theme
  - Focus on responsive design
  - A popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web

jQuery JavaScript library

- W3C DOM API is tedious and verbose
- jQuery is succinct and convenient
- Defines a global variable $ (or jQuery)

```
1  var domElem = document.getElementById("para");
2  domElem.setAttribute('style', 'color: red;');
3  $("#para").css("color", "red");
```

jQuery

- Let’s play
  - JSFiddle: [http://jsfiddle.net/](http://jsfiddle.net/)
  - Don’t forget to View Source on the Result frame

JSFiddle

```
<svg width="200" height="200">
  <rect x="0" y="0" width="200" height="200" fill="#FF0000" stroke="#00FF00" stroke-width="12.5"/>
  <title>Interactive SVG</title>
</svg>
```

SVG

- Scalable Vector Graphics (SVG)
  - Vector graphics in XML
  - Specification: [http://www.w3.org/TR/SVG/Overview.html](http://www.w3.org/TR/SVG/Overview.html)
Web graphics

- Great graphics = HTML + CSS + SVG
  - View in any browser, always sharp, quick to load
  - But static!!
- Great visualisation needs interactivity
- Great visualisations = HTML + CSS + SVG + JavaScript (D3.js)

Data-Driven Documents

- D3.js JavaScript library: http://d3js.org/
- Data + HTML + CSS + SVG + JavaScript
- Great Visualisation

D3.js

- In particular:
  - Three Little Circles http://bost.ocks.org/mike/circles/

Putting Everything Together

- Mash: A “supported” Web application
  - https://github.com/mlawry/unsw-cse-seng-mash-webapp
  - Study it in your own time
  - Ask me any questions about it via email
  - Bug fixes welcome (submit issues via GitHub)
- Mash uses:
  - Client-side: HTML + CSS + SVG + JavaScript + Ajax (D3.js, jQuery, Bootstrap)
  - Server-side: Node.js
  - Use it as a template for your project!
Some hints on starting your project

Node.js tooling
- Node.js on (recent versions of) Linux
  - Install using system package manager
    - Debian/Ubuntu: nodejs, npm
  - Use command line and a text editor
  - Integrated Development Environments
    - Eclipse (search “eclipse node.js”)
    - NetBeans (search “netbeans node.js”)

Node.js tutorials (Linux)
- Netbeans Nodejs Plugin: https://github.com/timboudreau/nb-nodejs
- Command line: http://blog.modulus.io/absolute-beginners-guide-to-nodejs

Node.js for Windows users
- Option 1: Switch to Linux?
- Option 2: Use Eclipse or NetBeans?
- Option 3: Use Node.js Tools for Visual Studio
  - Visual Studio 2013 Community Edition
  - Node.js Tools for Visual Studio
  - Node.js for Windows (MSI 64-bit)

Node.js tutorials (Windows)
  - Includes a demo on deployment to Azure (if you want to use it)

Tutorial on Express framework
- Express via the command line
  https://codeforgeek.com/2014/10/express-complete-tutorial-part-1/
- Deploy node.js apps to Heroku (free hosting service – conditions apply)
  https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction
Some UI examples


Thank You

Questions?

Email me: l.yao@unsw.edu.au

Remember
When learning coding, be brave

Acknowledgments
Lecture notes from Kenny Sahir