CSE 115
Introduction to Computer Science I
Announcements

Dr. Alphonce's e-mail glitched: not all unread e-mails were shown.

Please be patient while he catches up.
Announcements

A sample midterm exam will be posted early next week.

We will run midterm review sessions – details TBA.
Announcements

We are adding more UTA office hours – some new hours coming on-line tomorrow.

Lab Activity 03: most UTAs finished their grading, a few did not. Grading should be wrapped up Monday.
Announcements

Please complete the official Mid-semester Course Evaluation!
Road map

▶︎ review ◀

JavaScript Libraries

Plotly
Front End JavaScript

```html
<html>
<head></head>
<body>
<h1>First Web Page</h1>
<p>My content</p>
<div id="myDiv"></div>
<script src="myCode.js"></script>
</body>
</html>

```var myDiv = document.getElementById("myDiv"); myDiv.innerHTML = "Content added from JavaScript";```
<html>
<head></head>
<body>
<h1>First Web Page</h1>
<p>My content</p>
<div id="myDiv"></div>
<script src="myCode.js"></script>
</body>
</html>

HTML is a markup language
It has element defined by tags
Elements can have properties defined in their open tag as key-value pairs
Using a script element with an src property allows us to “import” javascript code
Front End JavaScript

```html
<html>
<head></head>

<body>
<h1>First Web Page</h1>
<p>My content</p>
<div id="myDiv"></div>
<script src="myCode.js"></script>
</body>

</html>

var myDiv = document.getElementById("myDiv");
myDiv.innerHTML = "Content added from JavaScript";
```
Front End JavaScript

```html
<html>
<head></head>
<body>
<h1>First Web Page</h1>
<p>My content</p>
<div id="myDiv"></div>
<script src="myCode.js"></script>
</body>
</html>

var myDiv = document.getElementById("myDiv");
myDiv.innerHTML = "Content added from JavaScript";
```
Road map

- review

  - JavaScript Libraries

  - Plotly
Download external libraries using the script tag the same way we downloaded our own code

Makes an HTTP request to download the library

We can add the script element in the head since we don't need any HTML elements rendered before downloading the library
When the page loads there will be a second request and the library is downloaded.

The library is simply a JavaScript file that others have written and shared with us for free. We call this open-source since everyone has access to the source code.

source: https://github.com/moment/moment/
JavaScript Libraries

We literally download their code and run it in our browser

```
// moment.js

(function (global, factory) {
  typeof exports === 'object' && typeof module !== 'undefined' ? module.exports = factory() :
  typeof define === 'function' && define.amd ? define(factory) :
  global момент = factory()

})(this, (function () {'use strict';

  var hookCallback;

  function hooks () {
    return hookCallback.apply(null, arguments);
  }

  // This is done to register the method called with moment()
  // without creating circular dependencies.
  function setHookCallback (callback) {
    hookCallback = callback;
  }

  function isArray(input) {
    return input instanceof Array || Object.prototype.toString.call(input) === '[object Array]';
  }

  function isObject(input) {
    // IEB will treat undefined and null as object if it wasn't for
    // input !== null
  }
```
JavaScript Libraries

Great!.. so what did we just download?
The library defines a function named `moment()` which returns the current time as an object that contains several other functions.

Here we get the current time and call the `format` function to choose how it's displayed using date/time placeholders (ex. MM for a 2 digit month).
JavaScript Libraries

```javascript
var elem = document.getElementById("datetime");
elem.innerHTML = moment().format("YYYY-MM-DD<br/><b>h:mm:ss a</b>"),
```

Displays the time in the `<p>` element with id "datetime" in the format

2018-10-18
10:23:57 pm
This is ok, but it'd be more useful to show the current time instead of the time our code ran

To do this, we'll first wrap our previous code in a function

Then we'll call the built-in "setInterval" function with our function as an argument (yes, you can do that!) which will call our function at fixed intervals

The second argument is a Number representing a time in milliseconds

This will call displayTime once per second to give us a live clock

```
function displayTime(){
    var elem = document.getElementById("datetime");
    elem.innerHTML = moment().format("YYYY-MM-DD<br/>\<b\>h:mm:ss a</b\>"的习惯);
}

setInterval(displayTime, 1000);
```
JavaScript Libraries

Reminder: You are learning concepts, not memorizing specific examples

The concept is using JavaScript libraries to do cool things without writing code from scratch

The moment library was an example of this concept
Road map

review

JavaScript Libraries

▶ Plotly ◀
Plotly Library

A JavaScript library that helps us generate plots and graphs

https://plot.ly/javascript/

*There is also a Python version of this library*
Plotly Library

We can now use Plotly in our code (plots.js)
Plotly Library

var data = [{
  x: [1930, 1940, 1950, 1960],
  y: [573076, 575901, 580132, 532759]
}];

Plotly.newPlot('myPlot', data);

Let's use plotly to make a line graph

The library has a function named newPlot that takes the id of an HTML element (the element should be a div) and an array as parameters

To make a line graph the array will contain an object(s) with "x" and "y" as keys
Plotly Library

```javascript
var data = [{
    x: [1930, 1940, 1950, 1960],
    y: [573076, 575901, 580132, 532759]
}];

Plotly.newPlot('myPlot', data);
```
var data = [{
    x: [1930, 1940, 1950, 1960],
    y: [573076, 575901, 580132, 532759],
    name: "Buffalo Population"
}];

var layout = {
    "title": "Buffalo Population",
    xaxis: {
        "title": "year"
    },
    yaxis: {
        "title": "population"
    }
}

Plotly.newPlot('myPlot', data, layout);

Data objects can have additional key-value pairs to add optional settings

Plotly.newPlot can take a 3rd optional parameter that is an object with layout settings
var data = [
    {
      x: [1930, 1940, 1950, 1960],
      y: [573076, 575901, 580132, 532759],
      name: "Buffalo Population"
    }
];

var layout = {
    title: "Buffalo Population",
    xaxis: {
      title: "year"
    },
    yaxis:
    {
      title: "population"
    }
};

Plotly.newPlot('myPlot', data, layout);
JavaScript Libraries

Reminder: You are learning concepts, not memorizing specific examples

We're showing you how to make line graphs in lecture and maps in the project

You are expected to be able to generate different types of graphs by reading the Plotly documentation