CS 4220 - NODE.JS & ANGULAR.JS

INTRODUCTION TO ANGULAR.JS

AGENDA

- Review Lab Assignment
- Data Binding
- Modules
- Controllers
- Scopes
- Views

LAB

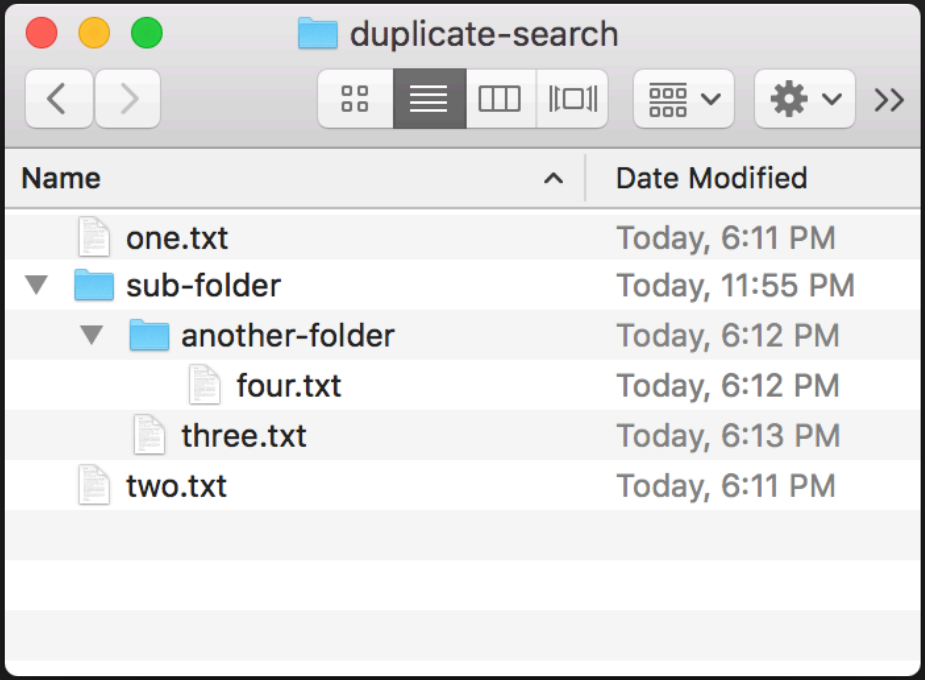
- Write a node.js application that:
  - Takes a directory as a command-line-argument
  - Traverses the directory
  - Traverses all sub-directories
  - Prints out all filenames (full-path) that are duplicates of each other.

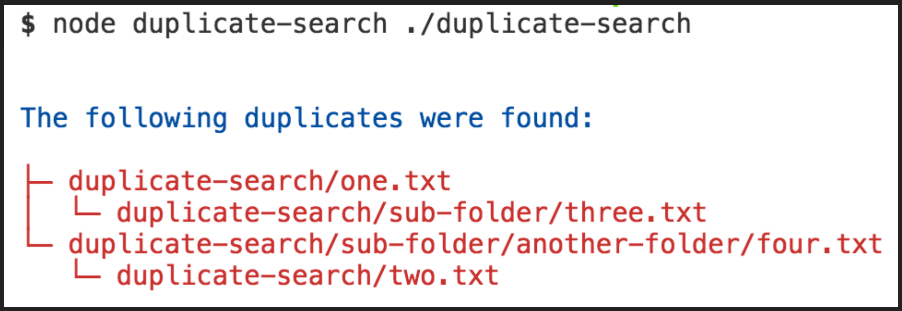
LAB – HINTS

- node-dir
    - Consider using a module like *node-dir* to traverse a folder structure
    - The *readFiles* method is particularly useful
    - Install: `node install node-dir`
    - URL: https://www.npmjs.com/package/node-dir
- treeify
    - Consider nicely formatted output as a nice-to-have. However, consider using a module like *treeify* to format the output in a meaningful way.
    - Install: node install treeify
    - URL: https://www.npmjs.com/package/treeify

LAB – SAMPLE RUN

- My *duplicate-search* folder contains files and a directory.
- The directory contains additional files and a directory.
- I run my *duplicate-search.js* application from the *parent directory*, and I pass in the *./duplicate-search* path.
- The output is displayed in a tree-like fashion.





```
$ node duplicate-search ./duplicate-search


The following duplicates were found:

├─ duplicate-search/one.txt
│  └─ duplicate-search/sub-folder/three.txt
└─ duplicate-search/sub-folder/another-folder/four.txt
   └─ duplicate-search/two.txt
```

DUE TODAY

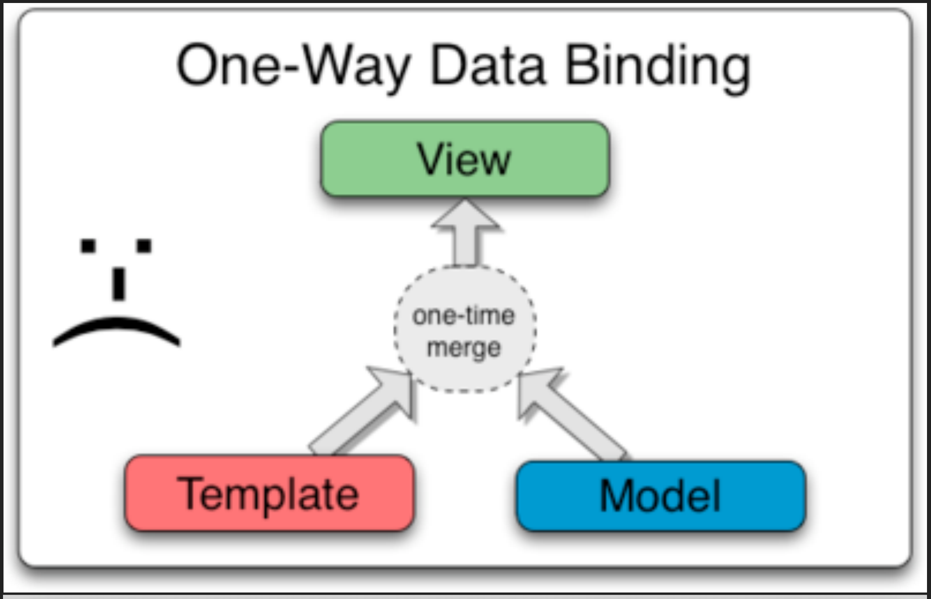
ANGULAR.JS

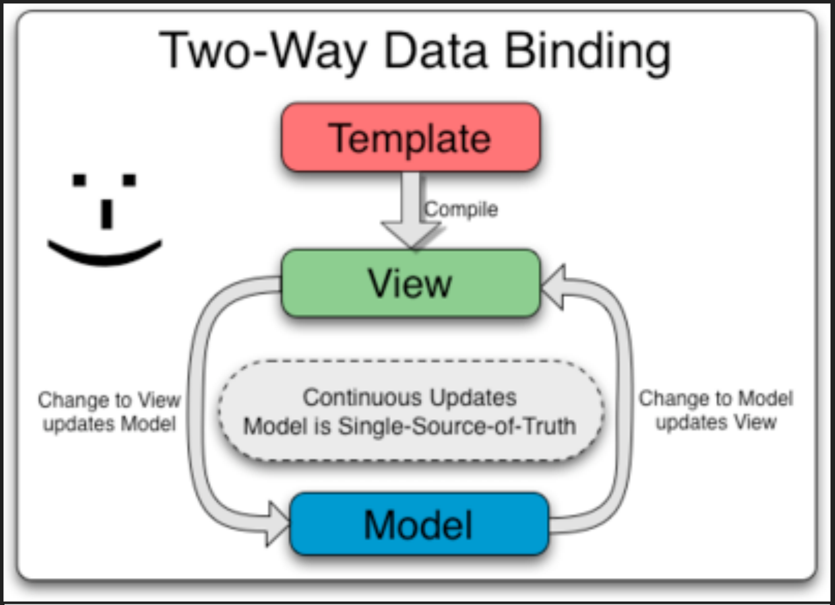
INTRODUCTION TO ANGULAR.JS

- Developer Guide
- AngularJS is a structural framework for dynamic web apps.
- AngularJS teaches the browser new syntax through a construct known as *directives*. It allows developers to use, and extend, HTML as the template language.
- AngularJS strives to eliminate the need for:
    - Registering callbacks
    - Manipulating HTML DOM programmatically
    - Marshaling data to and from the UI
    - Writing tons of initialization code just to get started

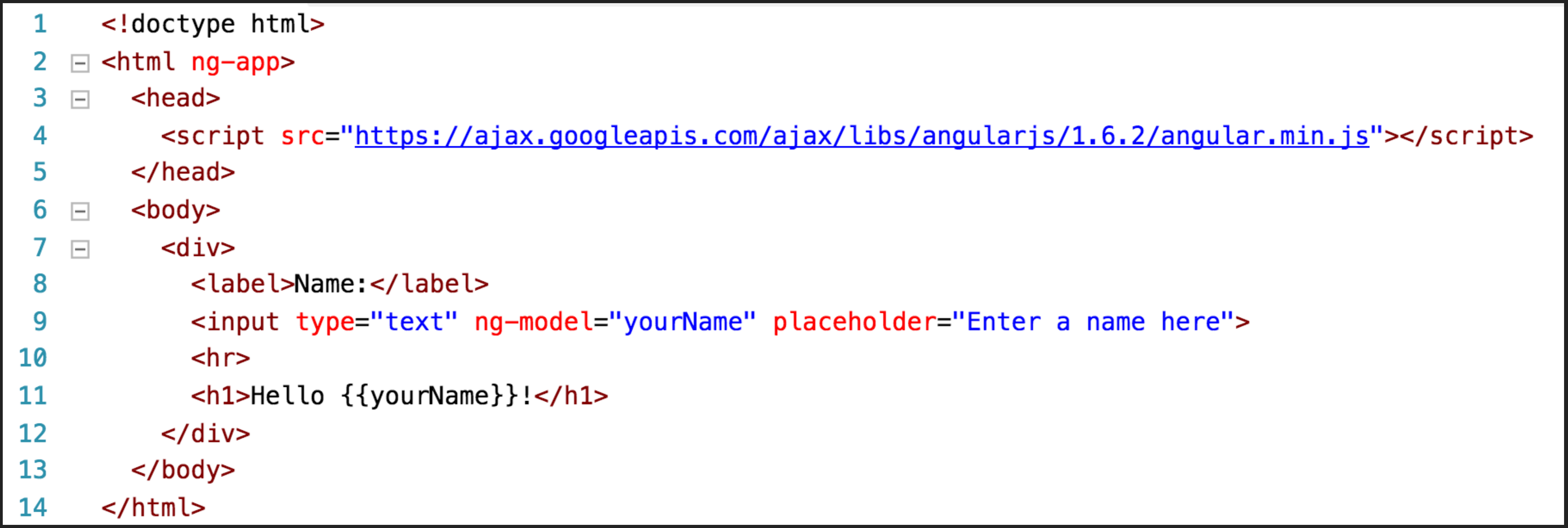
DATA BINDING

- Data-binding in AngularJS apps is the automatic synchronization of data between the model and view components.




Two-Way Data Binding
Template
Compile
View
Change to View updates Model
Continuous Updates
Model is Single-Source-of-Truth
Change to Model updates View
Model

DATA BINDING

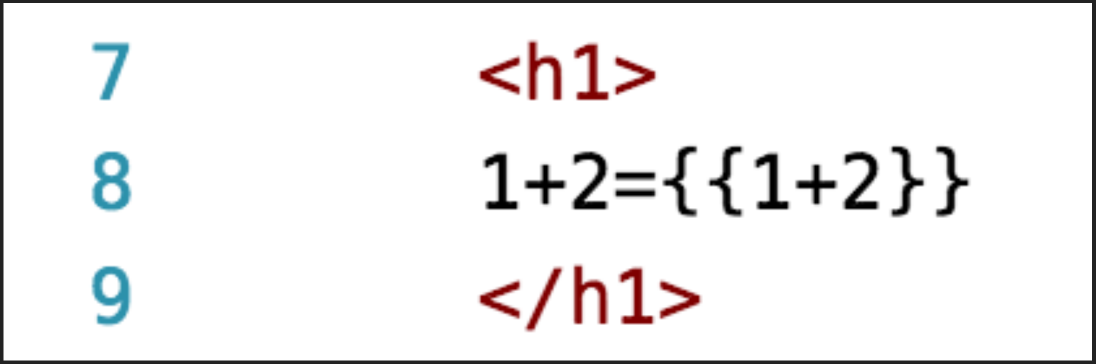


```html
<!doctype html>
<html ng-app>
  <head>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.2/angular.min.js"></script>
  </head>
  <body>
    <div>
      <label>Name:</label>
      <input type="text" ng-model="yourName" placeholder="Enter a name here">
      <hr>
      <h1>Hello {{yourName}}!</h1>
    </div>
  </body>
</html>
```

AngularJS - The Basics

EXPRESSIONS

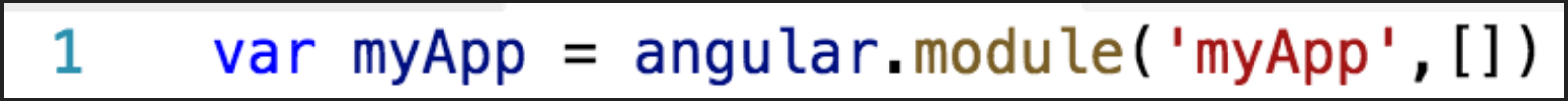
- JavaScript-like code snippets placed in interpolation bindings.



```
<h1>
1+2={{1+2}}
</h1>
```

MODULES

- In AngularJS, modules are containers for the different parts of your application.
  - Controllers, Services, Filters, etc…
- You can think of your application as the main module.



```
var myApp = angular.module('myApp',[])
```

CONTROLLERS

- A controller is defined by a JavaScript constructor function that is used to augment the AngularJS Scope.
- We use controllers to:
  - Set up the initial state of the $scope object.
  - Add behavior to the $scope object.

SCOPES

- Scope is an object that refers to the application model.
- It is an execution context for expressions.
- Scopes are arranged in hierarchical structure which mimic the DOM structure of the application.
- Scopes can watch expressions and propagate events.

TEMPLATES & VIEWS

- Templates looks like normal HTML, with some new markup.
- Angular parses and processes the template using the compiler.
- The loaded, transformed and rendered DOM is then called the view.