JavaScript

... high-level, dynamic, untyped, and interpreted programming language
... is prototype-based with first-class functions, …
... supporting object-oriented, imperative, and functional programming
... has an API for working with text, arrays, dates and regular expressions
All var declarations hoisted to top of
scope:
String: Lots of useful methods: indexOf(), charAt(), match(), search(),
replace(), toUpperCase(), toLowerCase(), slice(), substr(), ...
<script type="text/javascript" src="code.js"></script>
Javascript has the notion of a prototype object for each object instance
○ Prototype objects can have prototype objects forming a prototype chain
● On an object property read access JavaScript will search the up the prototype
chain until the property is found
● Effectively the properties of an object are its own property in addition to all the properties up the prototype chain. This is called prototype-based inheritance.

JavaScript (cont)

● Property updates are different: always create property in object if not found
● Can lead to fun in AngularJS

CSS

position property
position: static; (default) - Position in document flow
position: relative; Position relative to default position via
top, right, bottom, and left properties
position: fixed; Position to a fixed location on the screen via
top, right, bottom, and left properties
position: absolute; Position relative to ancestor absolute element via
top, right, bottom, and left properties
Fixed position (0,0) is top left corner
display: none; - Element is not displayed and takes no space in layout
display: inline; - Element is treated as an inline element.
display: block; - Element is treated as an block element.
visibility: hidden; - Element is hidden but space still allocated.
visibility: visible; - Element is normally displayed

CSS (cont)

Inheritance
○ Some properties (e.g. font-size) are inherited from parent elements
○ Others (border, background) are not inherited.
● Multiple rule matches
○ General idea: most specific rule wins
Text1 span.test { color: green }
Text2 span { color: red }
<link rel="stylesheet" type="text/css" href="myStyles.css" />
A CSS breakpointis the term used to describe a system of CSS rules that alter the web app
based on display size. It is part of what is known as responsive design and uses mechanism
such as the @media CSS selector to make rules apply based on different screen sizes.
@media only screen and (max-width: 767px)
and (orientation: portrait) {
/ *portrait phones* / }

URLS

http://host.company.com:80/a/b/c.html?user=Alice&year=2008#p2
Scheme (http:): identifies protocol used to fetch the content.
Host name (//host.company.com): name of a machine to connect to.

URLS (cont)

Server's port number (80): allows multiple servers to run on the same machine.
Hierarchical portion (/a/b/c.html): used by server to find content.
Query parameters (?user=Alice&year=2008): provides additional parameters
Fragment (#p2): Have browser scroll page to fragment (html: p2 is anchor tag)
Full URL: 2009 News
Absolute URL: <a href="/stock/quote.html">
same as
http://www.xyz.com/stock/quote.html
Relative URL (intra-site links): <a href="2008/March.html">
same as
http://www.xyz.com/news/2008/March.html
Define an anchor point (a position that can be referenced with # notation):
<a name="sec3">
Go to a different place in the same page: <a href="#sec3">

Other

Event bubbling and capturing are two ways of event propagation in the HTML DOM API, when an event occurs in an element inside another element, and both elements have registered a handle for that event. The event propagation mode determines in which order the elements receive the event.

Other (cont)

With bubbling, the event is first captured and handled by the innermost element and then propagated to outer elements.
With capturing, the event is first captured by the outermost element and propagated to the inner elements.
Capturing is also called "trickling", which helps remember the propagation order:
target is element clicked and current target is
Controller
cs142App.controller('StatesController', ['$scope', function($scope) { }]);
Module
var cs142App = angular.module('cs142App', ['ngRoute']);
Config
cs142App.config(['$routeProvider', function($routeProvider) {
$routeProvider.
when('/example', {
templateUrl: 'components/example/exampleTemplate.html',
controller: 'ExampleController'
}).
when('/states', {
templateUrl: 'components/states/statesTemplate.html',

Other (cont)

controller: 'StatesController'
}).
otherwise({
redirectTo: '/example'
});
}]);

Angular.js Example

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

Angular

ScopeB's prototype points at ScopeA
● Mostly does what you want (all properties of A appear in B)
● Useful since scopes are frequently created (e.g. ng-repeat, etc.)
● $rootScope is parent of all
Two-way binding works by watching when expressions in view template
change and updating the corresponding part of the DOM.

Angular (cont)

● Angular add a watch for every variable or function in template expressions
● During the digest processing all watched expressions are compared to their
previously known value and if different the template is reprocessed and the
DOM update
Directives
● Angular preferred method for building reusable components
○ Package together HTML template and Controller and extend templating language.
○ Ng prefixed items in templates are directives
Services
● Used to provide code modules across view components
○ Example: shared JavaScript libraries
● Angular has many built-in services
○ Server communication (model fetching)
$http, $resource, $xhrFactory
○ Wrapping DOM access (used for testing mocks)
$location, $window, $document, $timeout, $interval
○ Useful JavaScript functionality
$animate, $sce, $log

Angular (cont)

○ Angular internal accesses $rootScope, $parse, $compile
Angular APIs
● ngRoute - Client-side URL routing and URL management
○ CS142 - Passing parameters to the views
● ngResource - REST API access
○ CS142 - Fetch models
● ngCookies - Cookie management and access
● ngAria - Support for people with disabilities (Accessible Rich Internet Applications)
● ngTouch - Support for mobile devices (ngSwipeLeft, ngSwipeRight, etc.)
● ngAnimate - Support for animations (CSS & JavaScript based)
● ngSanitize - Parse and manipulate HTML safely