



Shaping Up with Angular

Level 1: Getting Started



What you need to know

Must know

HTML & CSS
JavaScript

Nice to know

Automated Testing

BDD - Behavior Driven Development

TDD - Test Driven Development

etc

Not so important

jQuery
Ruby on Rails
Python, PHP, etc
Databases



If you're using JavaScript to create a dynamic website, Angular is a good choice.

- Angular helps you organize your JavaScript
- Angular helps create responsive (as in fast) websites.
- Angular plays well with jQuery
- Angular is easy to test



Traditional Page-Refresh



Web Server



URL Request to server

Response with Webpage & Assets

User clicks on link, new Request

Response with Webpage & Assets

HTML JavaScript





Browser loads up entire webpage.

HTML JavaScript

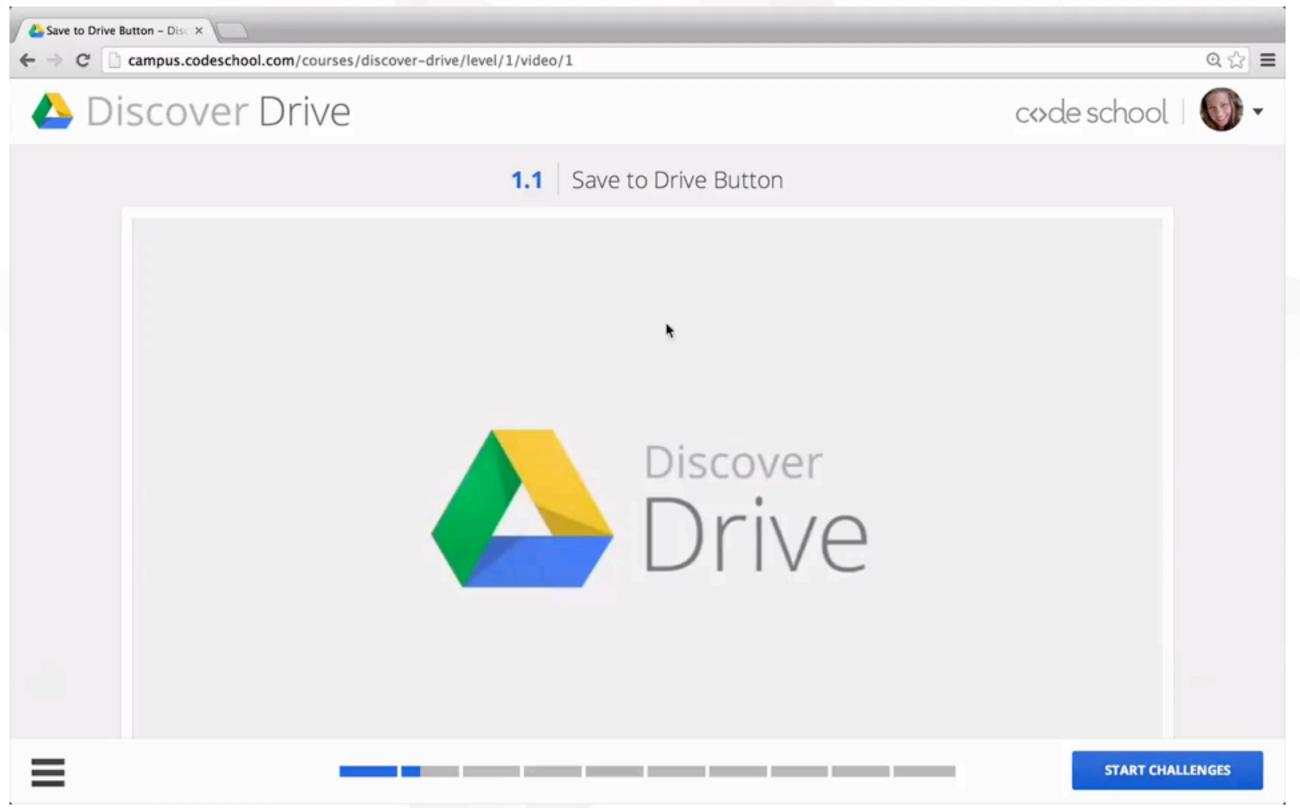




Browser loads up entire webpage.



A "responsive" website using Angular



Web Server



URL Request to server

Response with Webpage & Assets

User clicks on link, new Request

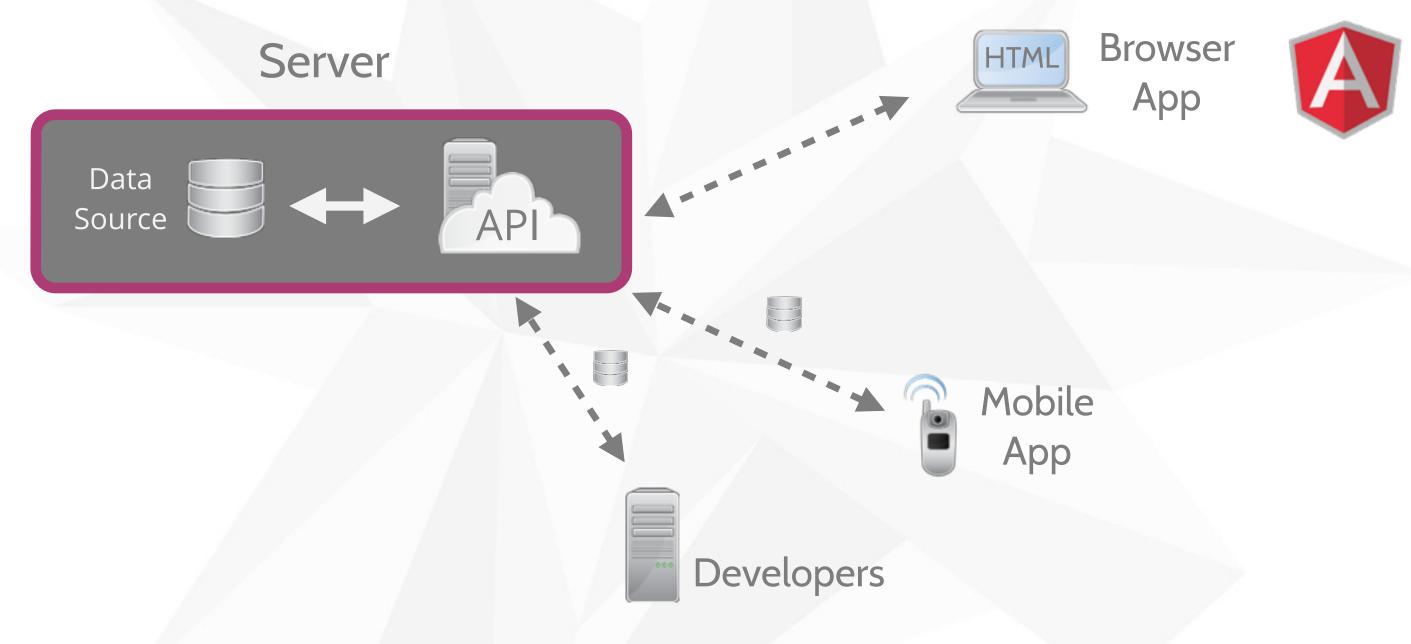
Response with JSON Data

HTML **JavaScript** Browser loads up entire webpage. DATA Data is loaded into

existing page.



Modern API-Driven Application





What is Angular JS?

A client-side JavaScript Framework for adding interactivity to HTML.

How do we tell our HTML when to trigger our JavaScript?

```
function Store(){
  alert('Welcome, Gregg!');
}

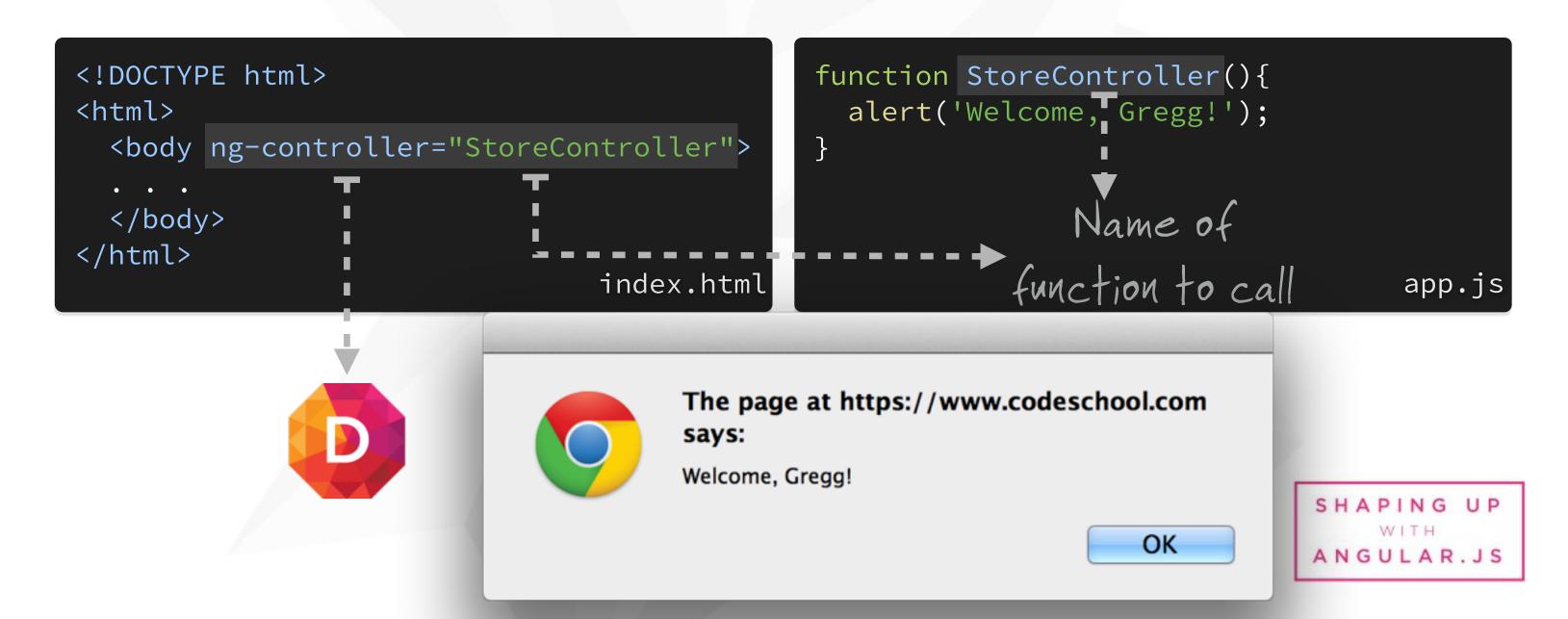
app.js
```

```
SHAPING UP
WITH
ANGULAR.JS
```



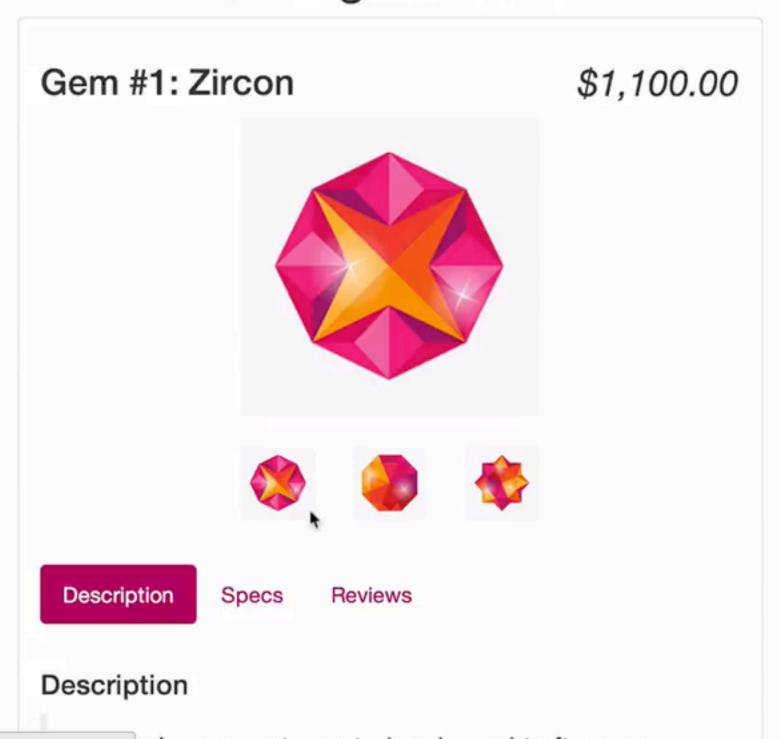
Directives

A Directive is a marker on a HTML tag that tells Angular to run or reference some JavaScript code.



Flatlander Crafted Gems

- an Angular store -





Downloading the libraries

Download AngularJS http://angularjs.org/

We'll need angular.min.js

Download Twitter Bootstrap http://getbootstrap.com/

We'll need bootstrap.min.css

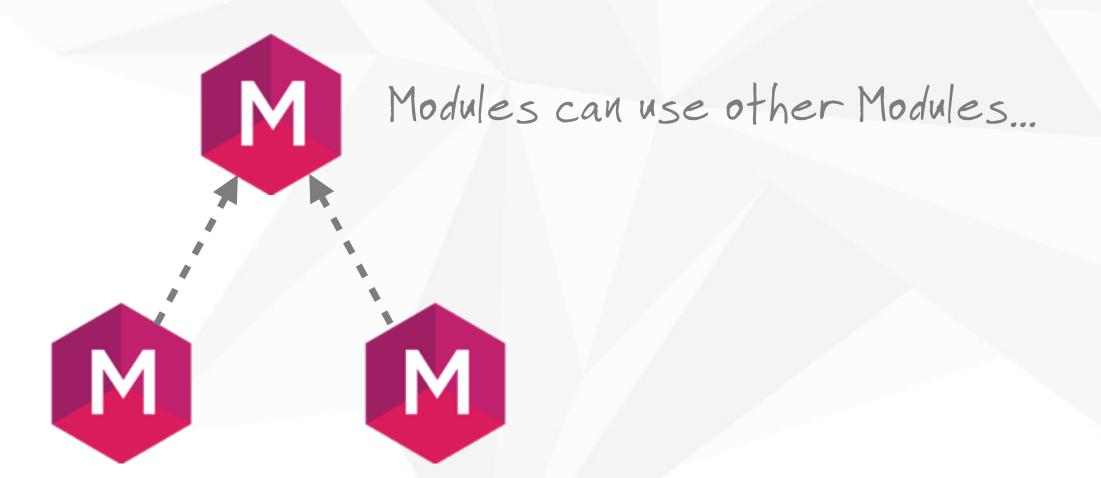


Getting Started



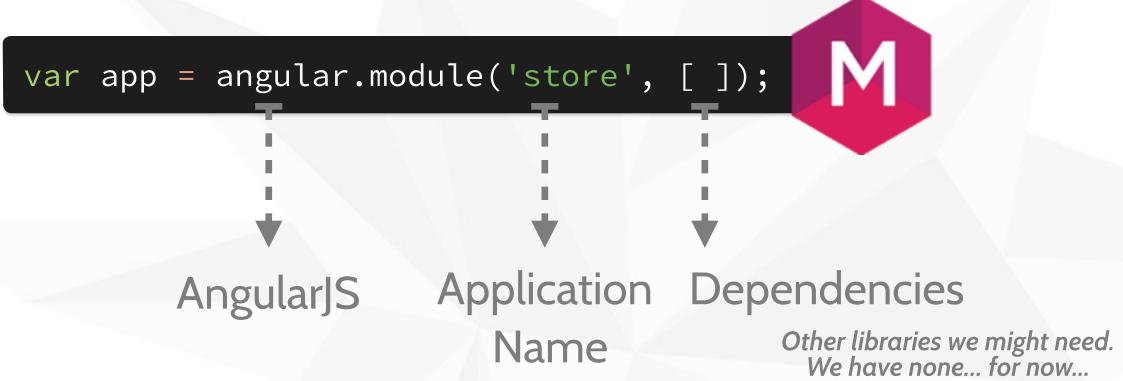
Modules

- Where we write pieces of our Angular application.
- Makes our code more maintainable, testable, and readable.
- Where we define dependencies for our app.





Creating Our First Module





Including Our Module

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="bootstrap.min.css" />
  </head>
  <body>
    <script type="text/javascript" src="angular.min.js"></script>
    <script type="text/javascript" src="app.js"></script>
  </body>
</html>
                                                               index.html
var app = angular.module('store', [ ]);
                                                 app.js
                                                                SHAPING UP
                                                                ANGULAR.JS
```



Including Our Module

```
Run this module
<!DOCTYPE html>
                                    when the document
<html ng-app="store"> -
 <head>
  </head>
 <body>
   <script type="text/javascript" src="angular.min.js"></script>
   <script type="text/javascript" src="app.js"></script>
 </body>
</html>
                                                index.html
```



Expressions

Allow you to insert dynamic values into your HTML.

Numerical Operations



String Operations



+ More Operations:

http://docs.angularjs.org/guide/expression



Including Our Module

```
<!DOCTYPE html>
<html ng-app="store">
  <head>
   <link rel="stylesheet" type="text/css" href="bootstrap.min.css" />
  </head>
  <body>
   <script type="text/javascript" src="angular.min.js"></script>
   <script type="text/javascript" src="app.js"></script>
   {{"hello" + " you"}}
 </body>
</html>
                                                          index.html
var app = angular.module('store', [ ]);
                                             app.js
 \Theta \Theta \Theta
         index.html
                     ×
                                                            SHAPING UP
 ANGULAR.JS
 hello you
```



Challenges



Working With Data

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
}
```

...just a simple object we want to print to the page.





Controllers

Controllers are where we define our app's behavior by defining functions and values.

```
Wrapping your Javascript in a closure is a good habit!
```

```
(function(){
  var app = angular.module('store', [ ]);

app.controller('StoreController', function(){
  });
});
app.js
```

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '...',
}
```

Notice that controller is attached to (inside) our app.

```
SHAPING UP
WITH
ANGULAR.JS
```



Storing Data Inside the Controller

```
(function(){
  var app = angular.module('store', [ ]);
 app.controller('StoreController', function(){
    this.product = gem;
 });
  var gem = {
    name: 'Dodecahedron',
    price: 2.95,
    description: '. . .',
})();
                                           app.js
```

Now how do we print out this data inside our webpage?



Our Current HTML

```
<!DOCTYPE html>
<html ng-app="store">
 <head>
   <link rel="stylesheet" type="text/css" href="bootstrap.min.css" />
 </head>
 <body>
   <div>
                                     Let's load our data into
     <h1> Product Name </h1>
     <h2> $Product Price </h2>
                                     this part of the page.
      Product Description 
   </div>
   <script type="text/javascript" src="angular.min.js"></script>
   <script type="text/javascript" src="app.js"></script>
 </body>
</html>
                                                              index.html
```



Attaching the Controller

```
(function(){
  var app = angular.module('store', [ ]);

app.controller('StoreController', function(){
    this.product = gem;
  });
  . . .
})();
```



Attaching the Controller

```
Directive Controller name
                                          Alias
<body>
  <div ng-controller="StoreController as store">
    <h1> Product Name </h1>
    <h2> $Product Price </h2>
     Product Description 
  </div>
  <script type="text/javascript" src="angular.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</body>
                                                        index.html
(function(){
 var app = angular.module('store', [ ]);
 app.controller('StoreController', function(){
   this.product = gem;
 });
})();
                                               app.js
```



Displaying Our First Product

```
(function(){
  var app = angular.module('store', [ ]);

app.controller('StoreController', function(){
    this.product = gem;
});
  . . .
})();
AngularJS

angular/rocks/mysocks.com

Dodecahedron

Some gems have hidden qualities beyond their luster, beyond their luster, beyond their shine... Dodeca is one of those gems.
```



Understanding Scope

```
<body>
  <div ng-controller="StoreController as store">
    <h1> {{store.product.name}} </h1>
    <h2> ${{store.product.price}} </h2>
     {{store.product.description}} 
  </div>
  {{store.product.name}}
  <script type="text/javascript" src="angular.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</body>
       Would never print a value!
```

The scope of the Controller is only inside here.



Challenges



Adding A Button

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '...',
}
```



Adding A Button

```
<body ng-controller="StoreController as store"> Directives to
the rescue!
    <h2> ${{store.product.price}} </h2>
     {{store.product.description}} 
    <button> Add to Cart </button> <-----</pre>
  </div>
  <script type="text/javascript" src="angular.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</body>
                                                       index.html
```

How can we only show this button ...

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
  canPurchase: false < = = ...when this is true?
}</pre>
```



NgShow Directive

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '...',
  canPurchase: false ***
}
```



Will only show the element if the value of the Expression is true.

```
10
      <script data-require="angular.js@1.2.x" src="http://code.angularjs.org/1.2.15/angul</pre>
      <script src="app.js"></script>
11
12
    </head>
13
14
15 -
    <body ng-controller="StoreController as store">
16
17
      <!-- Products Container
18 -
      <div class="list-group">
        <!-- Product Container -->
19
        <div class="list-group-item">
20 -
21
          <h1>{{store.product.name}}</h1>
22
          <h2>${{store.product.price}}</h2>
23
          {{store.product.description}}
          <button ng-show="store.product.canPurchase">Add to Cart
/button> T
24
25
        </div>
26
      </div>
27
    </body>
28
29
    </html>
                                                index.html
```

their shine... Dodeca is



NgHide Directive



NgHide Directive

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
  canPurchase: true,
  soldOut: true, ---- we want to hide it.
}
```



NgHide Directive

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
  canPurchase: true,
  soldOut: true, <---- we want to hide it.
}</pre>
```



Multiple Products

```
app.controller('StoreController', function(){
   this.product = gem;
});

var gem = {
   name: "Dodecahedron",
   price: 2.95,
   description: "...",
   canPurchase: true,
}
```



Multiple Products

```
app.controller('StoreController', function(){
  this.products = gems;
3); So we have multiple products...
var gems = [ ----- Now we have an array...
    name: "Dodecahedron",
    price: 2.95,
                                               Maybe a
Directive?
    description: ". . . ",
    canPurchase: true,
    name: "Pentagonal Gem",
    price: 5.95,
                                How might we display all these products in our template?
    description: ". . .",
    canPurchase: false,
  }...
                                                                         app.js
```







```
<body ng-controller="StoreController as store">
  <div>
                                                                   run.plnkr.co/nsIRASW7RSFzW3EH/
    <h1> {{store.products[0].name}} </h1>
                                                                   Dodecahedron
    <h2> ${{store.products[0].price}} </h2>
                                                                   Some gems have hidden qualities beyond their
     {{store.products[0].description}} 
                                                                   luster, beyond their shine... Dodeca is one of those $2.95
    <button ng-show="store.products[0].canPurchase">
                                                                   Add to Cart
       Add to Cart</button>
  </div>
                                                                   Pentagonal Gem
  <div>
                                                                   Origin of the Pentagonal Gem is unknown, hence
                                                                   its low value. It has a very high shine and 12 sides, $5.95
    <h1> {{store.products[1].name}} </h1>
    <h2> ${{store.products[1].price}} </h2>
     {{store.products[1].description}} 
    <button ng-show="store.products[1].canPurchase">
       Add to Cart</button>
                                                             Why... You get it.
  </div>
                            That works
</body>
                                                                                   index.html
```



```
<body ng-controller="StoreController as store">
  <div ng-repeat="product in store.products">
     <h1> {{product.name}} </h1>
                                                                          Repeat this section for
     <h2> ${{product.price}} </h2>
      {{product.description}} 
     <button ng-show="product.canPurchase">
       Add to Cart</button>
  </div>
                           \Theta \Theta \Theta
                                          AngularJS
</body>
                             run.plnkr.co/nslRASW7RSFzW3EH/
                                                                                    index.html
                            Dodecahedron
                            Some gems have hidden qualities beyond their
                            luster, beyond their shine... Dodeca is one of those $2.95
                            gems.
                             Add to Cart
                            Pentagonal Gem
```

Origin of the Pentagonal Gem is unknown, hence

however.

its low value. It has a very high shine and 12 sides, \$5.95



What We Have Learned So Far



Directives - HTML annotations that trigger Javascript behaviors



Modules - Where our application components live



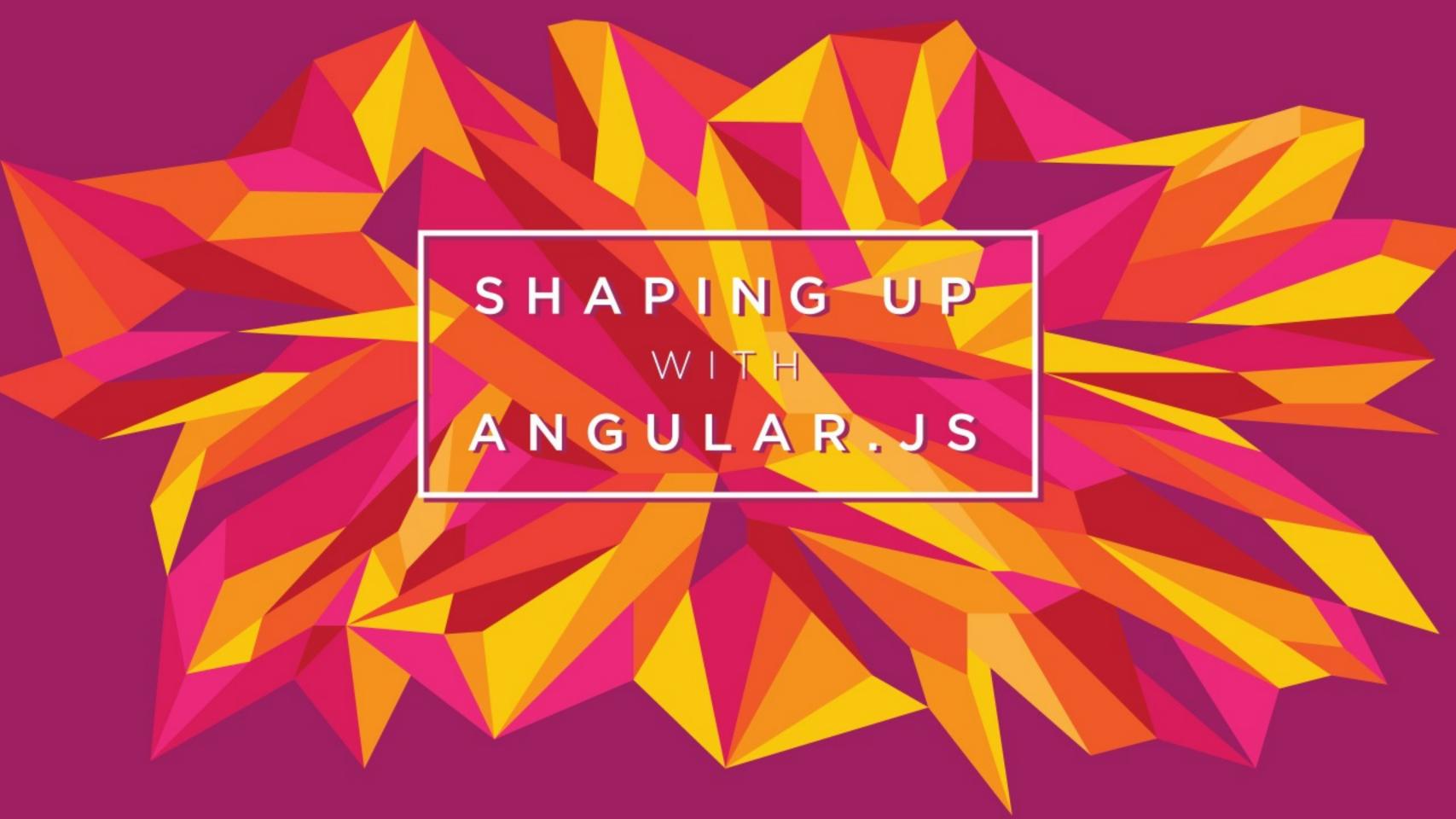
Controllers - Where we add application behavior

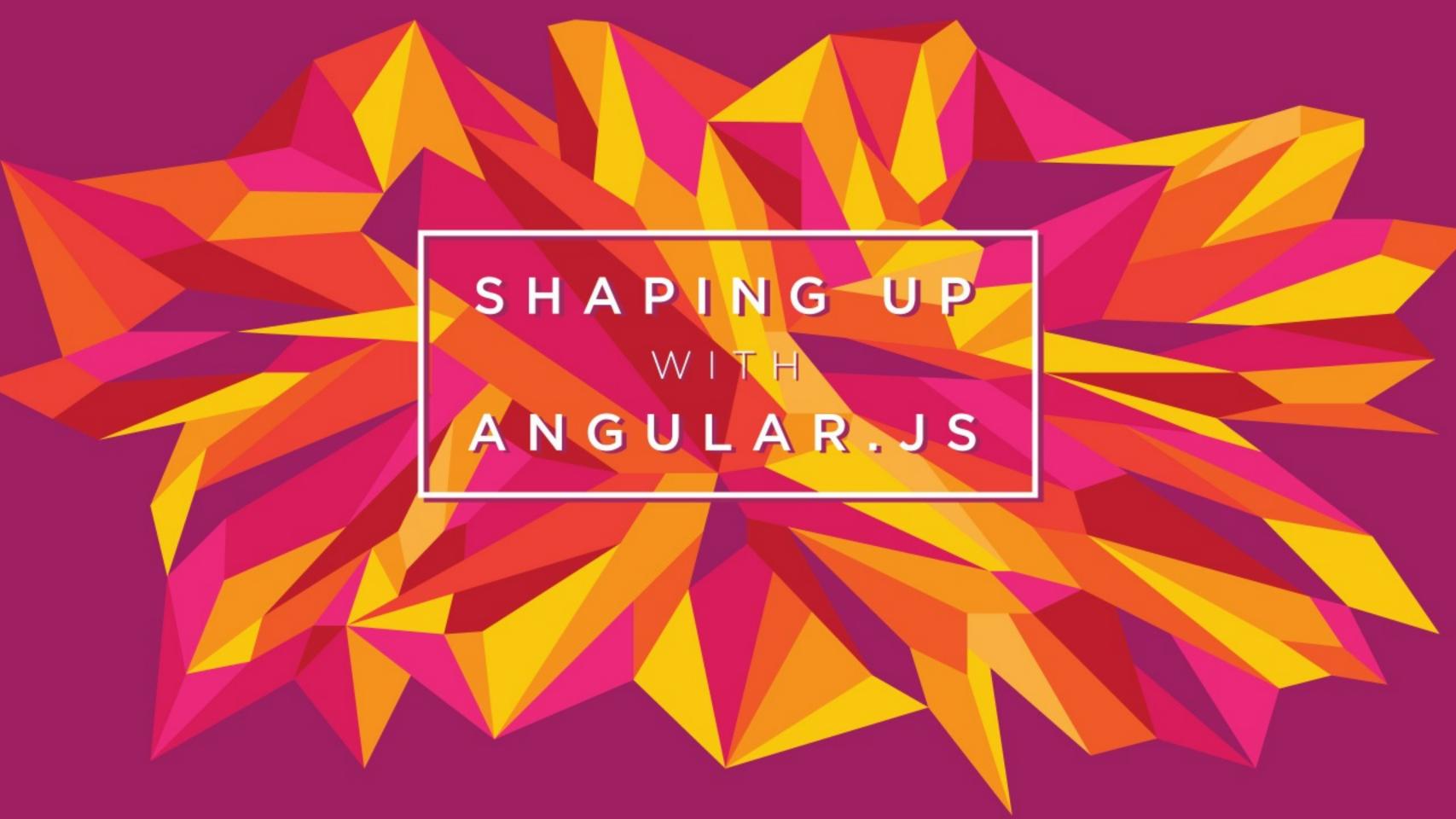


Expressions - How values get displayed within the page



Challenges







Shaping Up with Angular

Level 1: Getting Started



What you need to know

Must know

HTML & CSS
JavaScript

Nice to know

Automated Testing

BDD - Behavior Driven Development

TDD - Test Driven Development

etc

Not so important

jQuery
Ruby on Rails
Python, PHP, etc
Databases



If you're using JavaScript to create a dynamic website, Angular is a good choice.

- Angular helps you organize your JavaScript
- Angular helps create responsive (as in fast) websites.
- Angular plays well with jQuery
- Angular is easy to test



Traditional Page-Refresh



Web Server



URL Request to server

Response with Webpage & Assets

User clicks on link, new Request

Response with Webpage & Assets

HTML JavaScript





Browser loads up entire webpage.

HTML JavaScript

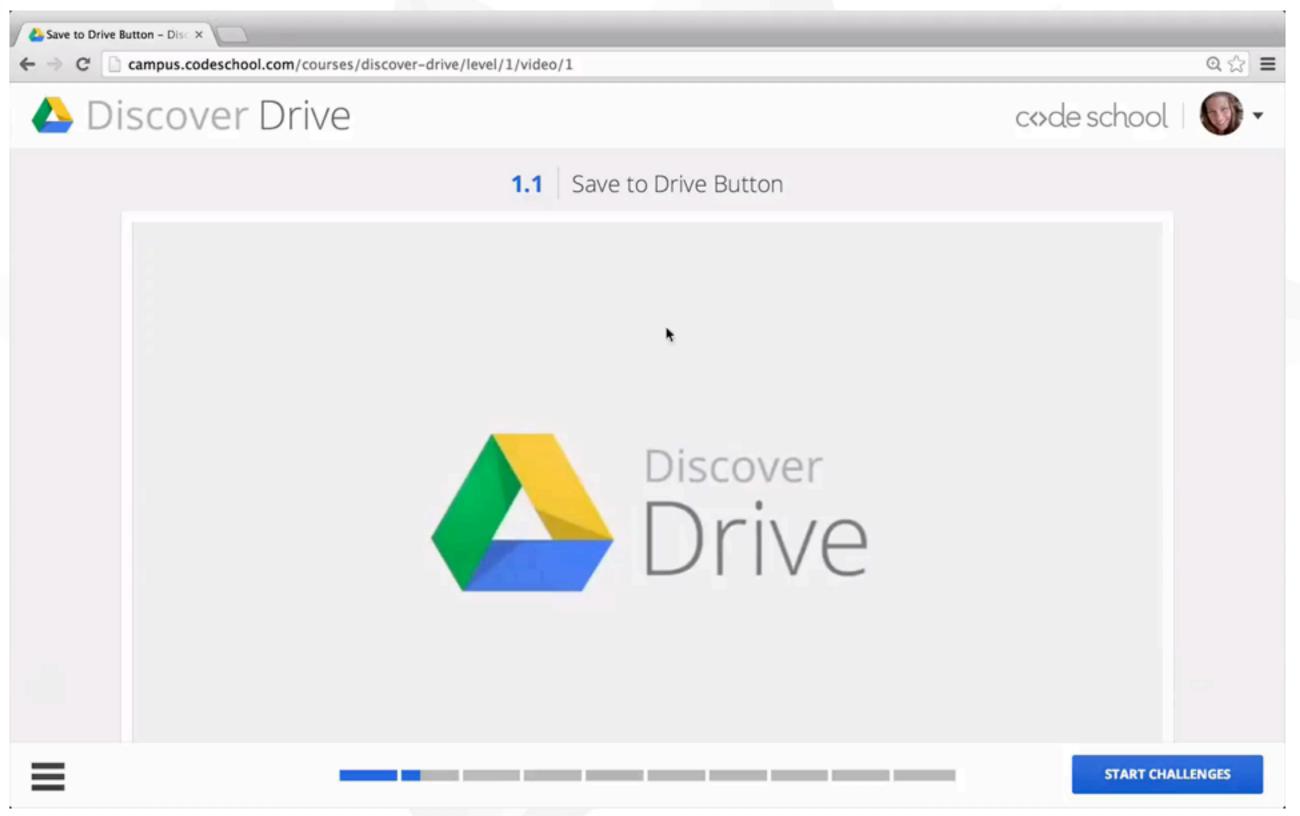




Browser loads up entire webpage.



A "responsive" website using Angular



Web Server



URL Request to server

Response with Webpage & Assets

User clicks on link, new Request

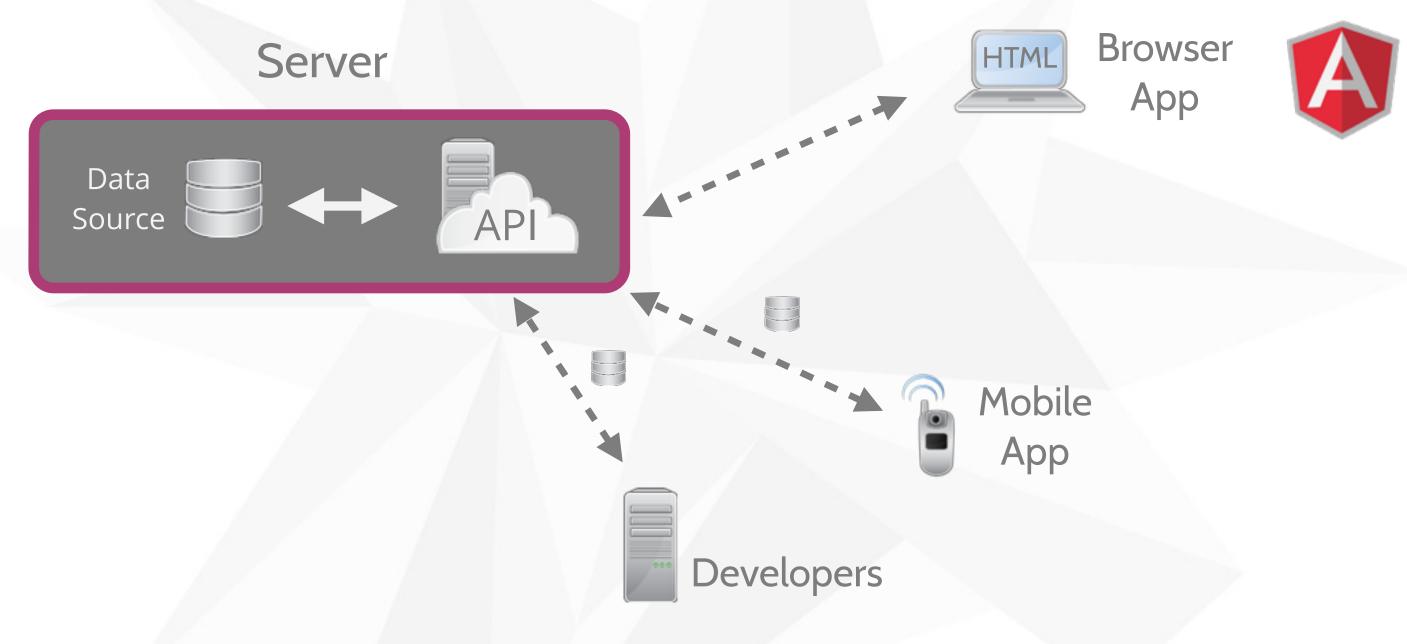
Response with JSON Data

HTML **JavaScript** Browser loads up entire webpage. DATA Data is loaded into

existing page.



Modern API-Driven Application





What is Angular JS?

A client-side JavaScript Framework for adding interactivity to HTML.

How do we tell our HTML when to trigger our JavaScript?

```
function Store(){
  alert('Welcome, Gregg!');
}

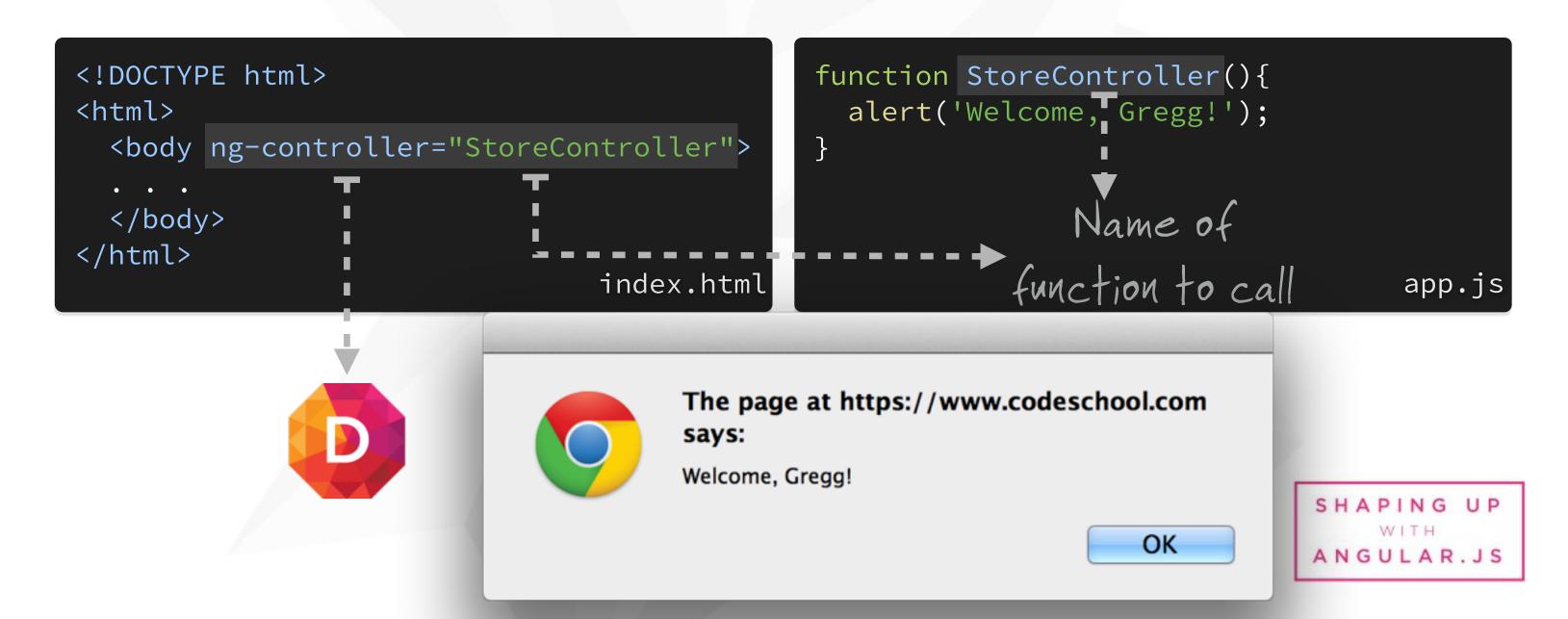
app.js
```

```
SHAPING UP
WITH
ANGULAR.JS
```



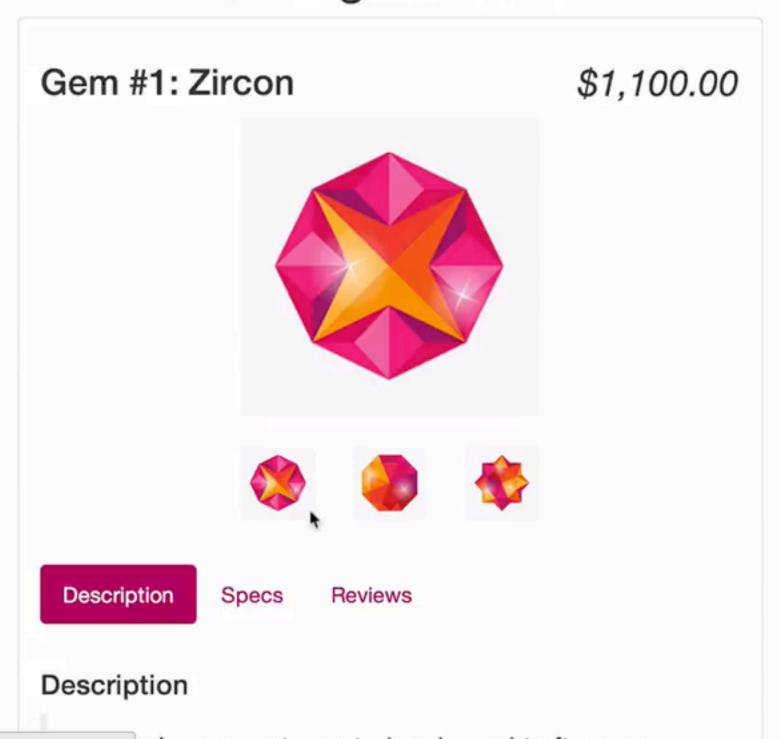
Directives

A Directive is a marker on a HTML tag that tells Angular to run or reference some JavaScript code.



Flatlander Crafted Gems

- an Angular store -





Downloading the libraries

Download AngularJS http://angularjs.org/

We'll need angular.min.js

Download Twitter Bootstrap http://getbootstrap.com/

We'll need bootstrap.min.css

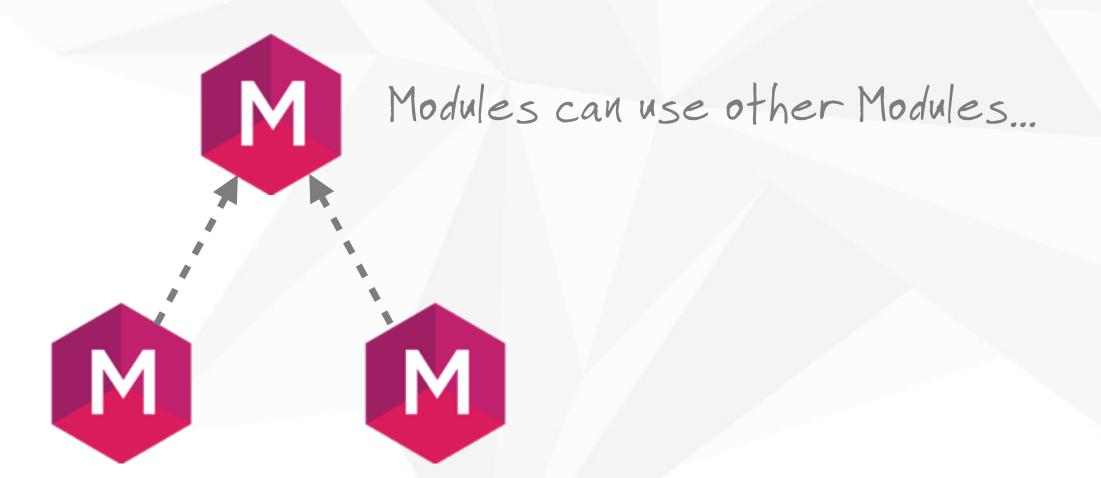


Getting Started



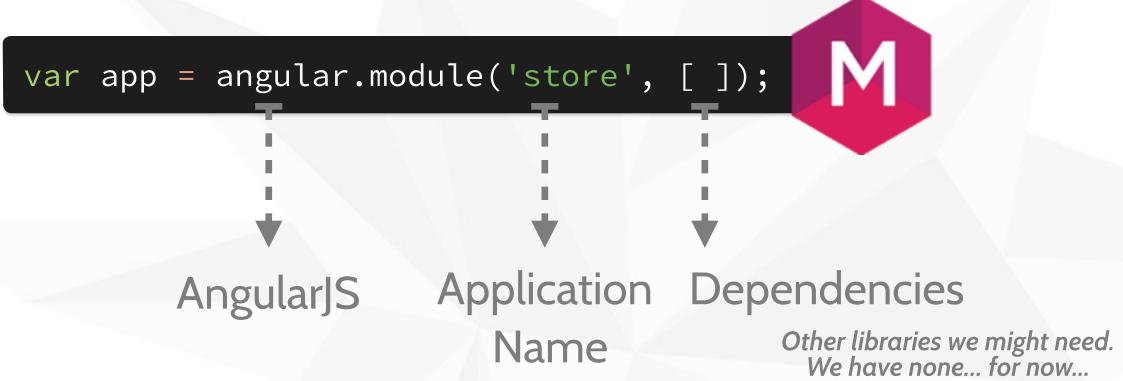
Modules

- Where we write pieces of our Angular application.
- Makes our code more maintainable, testable, and readable.
- Where we define dependencies for our app.





Creating Our First Module





Including Our Module

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="bootstrap.min.css" />
  </head>
  <body>
    <script type="text/javascript" src="angular.min.js"></script>
    <script type="text/javascript" src="app.js"></script>
  </body>
</html>
                                                               index.html
var app = angular.module('store', [ ]);
                                                 app.js
                                                                SHAPING UP
                                                                ANGULAR.JS
```



Including Our Module

```
Run this module
<!DOCTYPE html>
                                    when the document
<html ng-app="store"> -
 <head>
  </head>
 <body>
   <script type="text/javascript" src="angular.min.js"></script>
   <script type="text/javascript" src="app.js"></script>
 </body>
</html>
                                                index.html
```



Expressions

Allow you to insert dynamic values into your HTML.

Numerical Operations



String Operations



+ More Operations:

http://docs.angularjs.org/guide/expression



Including Our Module

```
<!DOCTYPE html>
<html ng-app="store">
  <head>
   <link rel="stylesheet" type="text/css" href="bootstrap.min.css" />
  </head>
  <body>
   <script type="text/javascript" src="angular.min.js"></script>
   <script type="text/javascript" src="app.js"></script>
   {{"hello" + " you"}}
 </body>
</html>
                                                          index.html
var app = angular.module('store', [ ]);
                                             app.js
 \Theta \Theta \Theta
         index.html
                     ×
                                                            SHAPING UP
 ANGULAR.JS
 hello you
```



Challenges



Working With Data

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
}
```

...just a simple object we want to print to the page.





Controllers

Controllers are where we define our app's behavior by defining functions and values.

```
Wrapping your Javascript in a closure is a good habit!
```

```
(function(){
  var app = angular.module('store', [ ]);

app.controller('StoreController', function(){
  });
});
app.js
```

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '...',
}
```

Notice that controller is attached to (inside) our app.

```
SHAPING UP
WITH
ANGULAR.JS
```



Storing Data Inside the Controller

```
(function(){
  var app = angular.module('store', [ ]);
 app.controller('StoreController', function(){
    this.product = gem;
 });
  var gem = {
    name: 'Dodecahedron',
    price: 2.95,
    description: '. . .',
})();
                                           app.js
```

Now how do we print out this data inside our webpage?



Our Current HTML

```
<!DOCTYPE html>
<html ng-app="store">
 <head>
   <link rel="stylesheet" type="text/css" href="bootstrap.min.css" />
 </head>
 <body>
   <div>
                                     Let's load our data into
     <h1> Product Name </h1>
     <h2> $Product Price </h2>
                                     this part of the page.
      Product Description 
   </div>
   <script type="text/javascript" src="angular.min.js"></script>
   <script type="text/javascript" src="app.js"></script>
 </body>
</html>
                                                              index.html
```



Attaching the Controller

```
(function(){
  var app = angular.module('store', [ ]);

app.controller('StoreController', function(){
    this.product = gem;
  });
  ...
})();
app.js
```



Attaching the Controller

```
Directive Controller name
                                          Alias
<body>
  <div ng-controller="StoreController as store">
    <h1> Product Name </h1>
    <h2> $Product Price </h2>
     Product Description 
  </div>
  <script type="text/javascript" src="angular.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</body>
                                                        index.html
(function(){
 var app = angular.module('store', [ ]);
 app.controller('StoreController', function(){
   this.product = gem;
 });
})();
                                               app.js
```



Displaying Our First Product

```
(function(){
  var app = angular.module('store', []);

app.controller('StoreController', function(){
  this.product = gem;
});
...
});
one of those gems.
Angular/S

angular/rocks/mysocks.com

Dodecahedron
Some gems have hidden qualities beyond their luster, beyond their shine... Dodeca is one of those gems.

$2.95
```



Understanding Scope

```
<body>
  <div ng-controller="StoreController as store">
    <h1> {{store.product.name}} </h1>
    <h2> ${{store.product.price}} </h2>
     {{store.product.description}} 
  </div>
  {{store.product.name}}
  <script type="text/javascript" src="angular.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</body>
       Would never print a value!
```

The scope of the Controller is only inside here.



Challenges



Adding A Button

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '...',
}
```



Adding A Button

```
<body ng-controller="StoreController as store"> Directives to
the rescue!
    <h2> ${{store.product.price}} </h2>
     {{store.product.description}} 
    <button> Add to Cart </button> <-----</pre>
  </div>
  <script type="text/javascript" src="angular.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</body>
                                                       index.html
```

How can we only show this button ...

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
  canPurchase: false < = = ...when this is true?
}</pre>
```



NgShow Directive

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '...',
  canPurchase: false ***
}
```



Will only show the element if the value of the Expression is true.

```
10
      <script data-require="angular.js@1.2.x" src="http://code.angularjs.org/1.2.15/angul</pre>
      <script src="app.js"></script>
11
12
    </head>
13
14
15 -
    <body ng-controller="StoreController as store">
16
17
      <!-- Products Container
18 -
      <div class="list-group">
        <!-- Product Container -->
19
        <div class="list-group-item">
20 -
21
          <h1>{{store.product.name}}</h1>
22
          <h2>${{store.product.price}}</h2>
23
          {{store.product.description}}
          <button ng-show="store.product.canPurchase">Add to Cart
/button> T
24
25
        </div>
26
      </div>
27
    </body>
28
29
    </html>
                                                index.html
```

their shine... Dodeca is



NgHide Directive



NgHide Directive

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
  canPurchase: true,
  soldOut: true, ---- we want to hide it.
}
```



NgHide Directive

```
var gem = {
  name: 'Dodecahedron',
  price: 2.95,
  description: '. . .',
  canPurchase: true,
  soldOut: true, <---- we want to hide it.
}</pre>
```



Multiple Products

```
app.controller('StoreController', function(){
   this.product = gem;
});

var gem = {
   name: "Dodecahedron",
   price: 2.95,
   description: "...",
   canPurchase: true,
}
```



Multiple Products

```
app.controller('StoreController', function(){
  this.products = gems;
}); So we have multiple products...
var gems = [ ----- Now we have an array...
    name: "Dodecahedron",
    price: 2.95,
                                               Maybe a
Directive?
    description: ". . . ",
    canPurchase: true,
    name: "Pentagonal Gem",
    price: 5.95,
                                How might we display all these products in our template?
    description: ". . .",
    canPurchase: false,
  }...
                                                                         app.js
```







```
<body ng-controller="StoreController as store">
  <div>
                                                                   run.plnkr.co/nsIRASW7RSFzW3EH/
    <h1> {{store.products[0].name}} </h1>
                                                                   Dodecahedron
    <h2> ${{store.products[0].price}} </h2>
                                                                   Some gems have hidden qualities beyond their
     {{store.products[0].description}} 
                                                                   luster, beyond their shine... Dodeca is one of those $2.95
    <button ng-show="store.products[0].canPurchase">
                                                                   Add to Cart
       Add to Cart</button>
  </div>
                                                                   Pentagonal Gem
  <div>
                                                                   Origin of the Pentagonal Gem is unknown, hence
                                                                   its low value. It has a very high shine and 12 sides, $5.95
    <h1> {{store.products[1].name}} </h1>
    <h2> ${{store.products[1].price}} </h2>
     {{store.products[1].description}} 
    <button ng-show="store.products[1].canPurchase">
       Add to Cart</button>
                                                             Why... You get it.
  </div>
                            That works
</body>
                                                                                   index.html
```



```
<body ng-controller="StoreController as store">
  <div ng-repeat="product in store.products">
     <h1> {{product.name}} </h1>
                                                                          Repeat this section for
     <h2> ${{product.price}} </h2>
      {{product.description}} 
     <button ng-show="product.canPurchase">
       Add to Cart</button>
  </div>
                           \Theta \Theta \Theta
                                          AngularJS
</body>
                             run.plnkr.co/nslRASW7RSFzW3EH/
                                                                                    index.html
                            Dodecahedron
                            Some gems have hidden qualities beyond their
                            luster, beyond their shine... Dodeca is one of those $2.95
                            gems.
                             Add to Cart
                            Pentagonal Gem
```

Origin of the Pentagonal Gem is unknown, hence

however.

its low value. It has a very high shine and 12 sides, \$5.95



What We Have Learned So Far



Directives - HTML annotations that trigger Javascript behaviors



Modules - Where our application components live



Controllers - Where we add application behavior



Expressions - How values get displayed within the page



Challenges

