AngularJS & Bootstrap

Tabs, Forms, Models

Tabs inside out

- Bootstrap has classes that make it easy to create tabs
- But how do we give behavior to these tabs?
- How tabs work:
  - If you select 1 tab,
  - it becomes active and
  - its content gets displayed on the page
- How do we select a tab?
  - using ng-click
  - ng-click takes expression that assigns a value to a variable, say tab.
  - tab == 1 when we click on the first, etc.

Example using tabs

```
<section ng-init="tab = 1”>
    <ul class="nav nav-pills">
        <li>
            <a href ng-click="tab=1"> Description </a>
        </li>
        <li>
            <a href ng-click="tab=2">Specs </a>
        </li>
        <li>
            <a href ng-click="tab=3">Reviews </a>
        </li>
     </ul>
     {{tab}} <!– debugging, display tab -->
</section>
```

2-way Data Binding

- When ***ng-click="tab = 1"*** changes the value of tab, the ***{{tab}}*** expression automatically gets updated

- This means expressions are ***re-evaluated*** when a property changes:
  - E.g., the value of ***tab*** changes
  - Expressions that refer to ***tab*** get re-evaluated when that happens

- Change to view updates model ***AND*** change to Model updates view

Clicking link updates value of tab

```
<section>
    <ul class="nav nav-pills">
        <li ng-class="{active:tab===1}">
            <a href ng-click="tab=1"> Description </a>
        </li>
        <li ng-class="{active:tab===2}">
            <a href ng-click="tab=2">Specs </a>
        </li>
        <li ng-class="{active:tab===3}">
            <a href ng-click="tab=3">Reviews </a>
        </li>
     </ul>
</section>
```

Changing tab triggers which panel to show

```
<div class="panel" ng-show="tab === 1">
    <h4>Description </h4>
    <p>{{product.description}}</p>
</div>
<div class="panel" ng-show="tab === 2">
   <h4>Specifications </h4>
   <blockquote>None yet</blockquote>
</div>
<div class="panel" ng-show="tab === 3">
   <h4>Reviews</h4>
   <blockquote>None yet</blockquote>
</div>
```

Setting initial values

- Use ***ng-init*** to set initial value of an expression

```
<section ng-init="tab = 1">
```

Changing tab updates <li> element

```
<section ng-init="tab = 1">
    <ul class="nav nav-pills">
        <li ng-class="{active:tab===1}">
            <a href ng-click="tab=1"> Description </a>
        </li>
        <li ng-class="{active:tab===2}">
            <a href ng-click="tab=2">Specs </a>
        </li>
        <li ng-class="{active:tab===3}">
            <a href ng-click="tab=3">Reviews </a>
        </li>
     </ul>
</section>
```

Cleaning dirty code

- Application's logic is inside our HTML.
  - Does not feel right
  - Makes code dirty
- Pull logic out of HTML to organize it
  - Maybe in a controller
  - Makes code look clean
  - Initialization, assignment, comparison code belong in controller

Forms and models

- Write code for basic html form
- Give form a name, will be useful
- Add code for live preview of form data
- Bind an object (model) to the form element using ***ng-model*** directive
- Use bootstrap css classes to embellish form
- Remember to submit the form

Binding Elements

- With Checkbox: bind the ng-model to the checkbox.

```
<input ng-model="review.terms"
type="checkbox" /> I agree to the terms.
```

- This sets value to true or false.

Binding Elements

- With Radio Buttons:

```
<input ng-model="review.color" type="radio"
value="red" /> Red
<input ng-model="review.color" type="radio"
value="blue" /> Blue
<input ng-model="review.color" type="radio"
value="green" /> Green
```

- Sets the proper value based on which is selected.

Submitting form

- Use ***ng-submit*** directive
  - allows us to call a function when the form is submitted.
- Define submission function in controller
  - Remember to initialize object that form will use
  - Can be an empty object
- Reset form to remove previous values
- Next time ➔ Form validation

Resources

- https://angularjs.org
- http://getbootstrap.com/css/#forms
- https://www.codeschool.com/paths/javascript