Don't Forget the Validation of Data

Paul D. Sheriff

PRESIDENT, PDSA, INC.

@pdsainc www.pdsa.com psheriff@pdsa.com

Goals

Angular validation basics

HTML changes for Angular validation

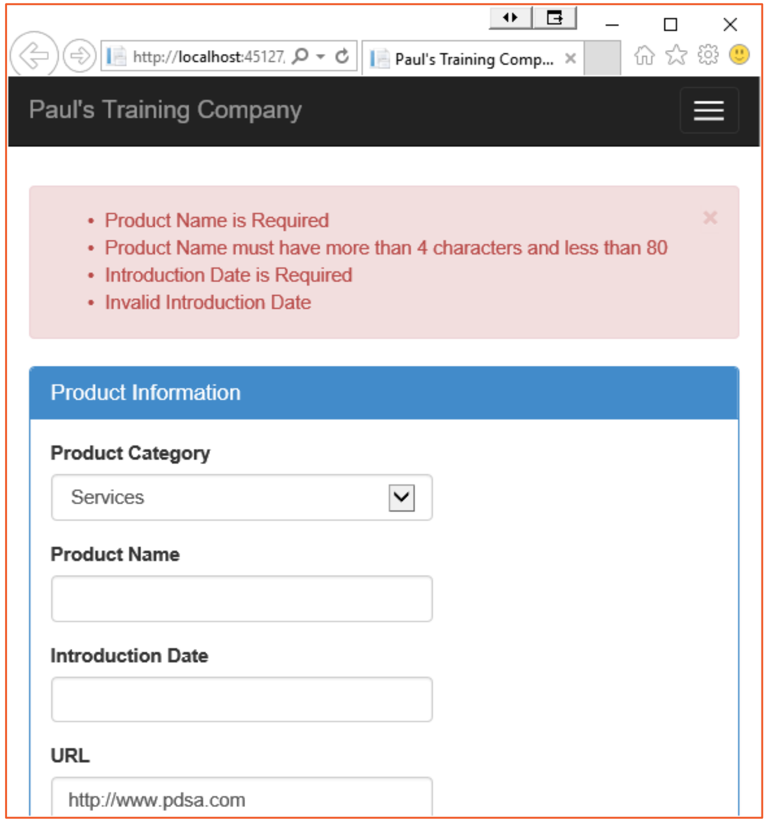
Controller changes for Angular validation

Server-side validation

Angular Validation Basics

Basics of Validation

- Create form tag with name
  - <form name="productForm" novalidate>
- Can check state of form
  - productForm.$valid
  - All controls are valid



- Product Name is Required
- Product Name must have more than 4 characters and less than 80
- Introduction Date is Required
- Invalid Introduction Date

Product Information

Product Category

Services

Product Name

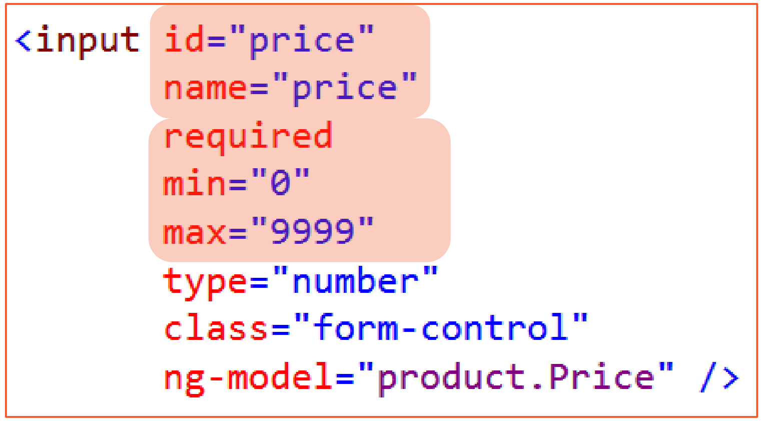
Introduction Date

URL

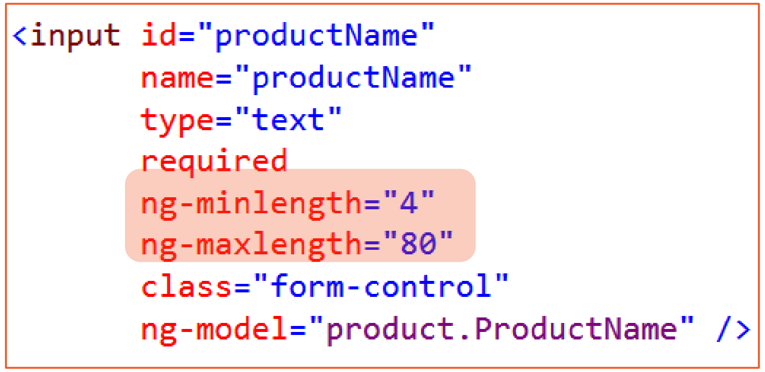
http://www.pdsa.com

Basics of Validation

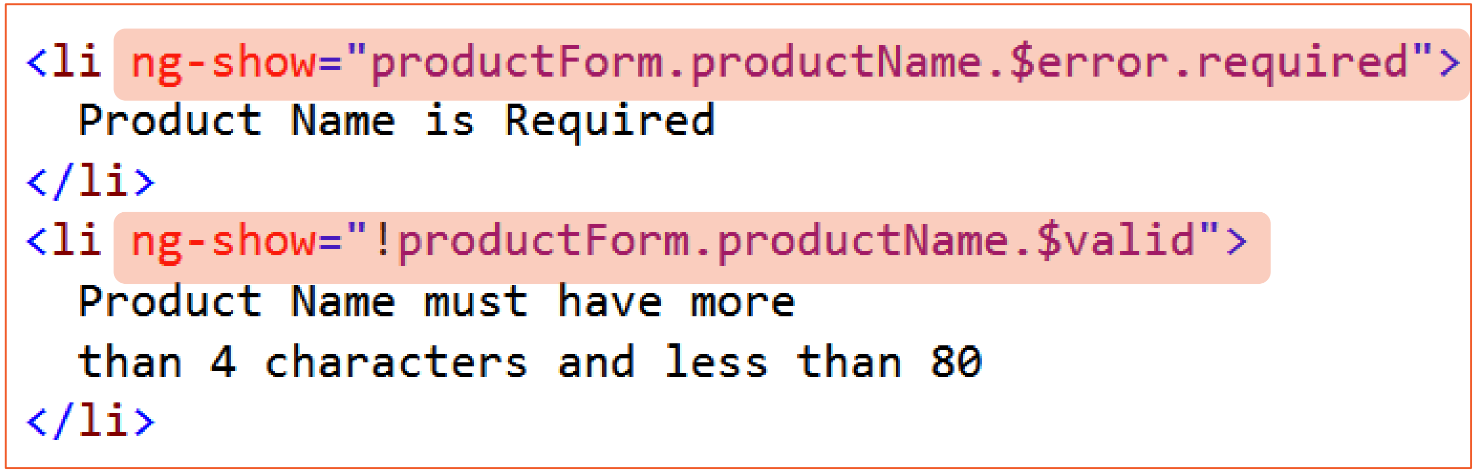
- Set name attribute
  - On all input fields you wish to validate
- Add HTML 5 attributes
  - required
  - min
  - max
- Add Angular attributes
  - ng-minlength
  - ng-maxlength



```
<input id="price"
       name="price"
       required
       min="0"
       max="9999"
       type="number"
       class="form-control"
       ng-model="product.Price" />
```



```
<input id="productName"
       name="productName"
       type="text"
       required
       ng-minlength="4"
       ng-maxlength="80"
       class="form-control"
       ng-model="product.ProductName" />
```



```
<li ng-show="productForm.productName.$error.required">
  Product Name is Required
</li>
<li ng-show="!productForm.productName.$valid">
  Product Name must have more
  than 4 characters and less than 80
</li>
```

Display validation messages

- Use ng-show
- Use the format:
  - <form name>.<control name>.$<property>

Demo

Modify HTML

Controller Changes to Support Validation

Modify Controller for Validation

Modify saveClick function

Check productForm.$valid

Set isValid value appropriately

$setPristine() to handle server-side validation

Write custom validation

Demo

Controller changes

Server-Side Validation

Don’t Rely on Client-Side Validation

Hackers can by-pass

Can’t do everything client-side

Need Server-Side Validation

Need to re-validate all data

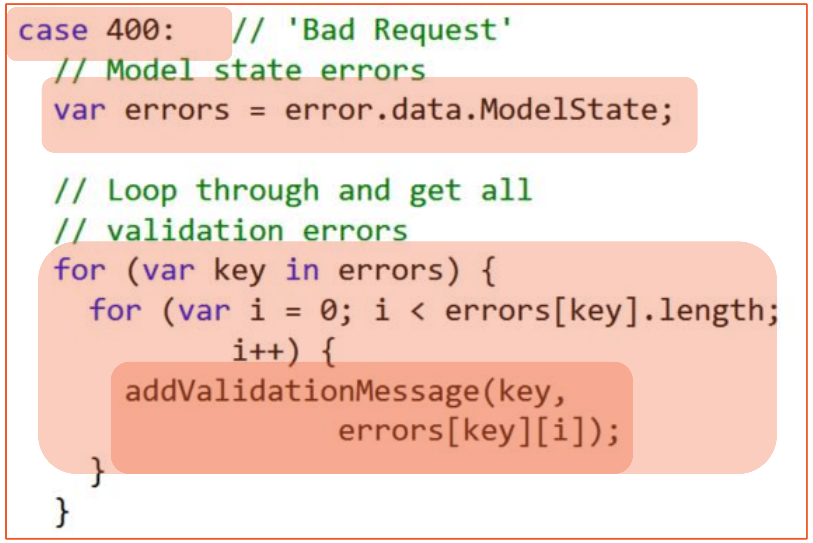
Use EF for validation

Convert EF Model State to Web API Model State

Return BadRequest()

More Business Rules

- Add 400 case in handleException
- Get ModelState from data property
- Loop through validation errors
- Add message to messages array



```
case 400:    // 'Bad Request'
  // Model state errors
  var errors = error.data.ModelState;

  // Loop through and get all
  // validation errors
  for (var key in errors) {
    for (var i = 0; i < errors[key].length;
                i++) {
      addValidationMessage(key,
                  errors[key][i]);
    }
  }
```

Demo

Server-side validation

Summary

Learned Angular validation

- HTML 5 validation
- ng-* validation

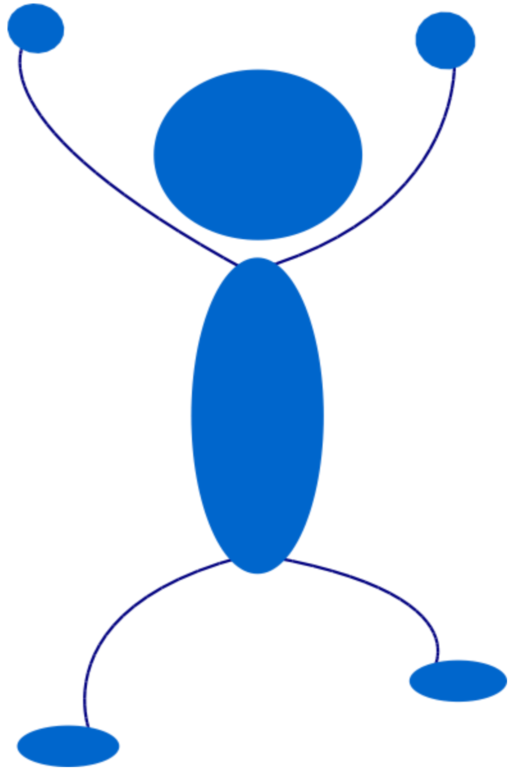
Additional validation on client-side

Server-side validation

- Communicating back to client

Course Summary

- Step-by-step we moved to Angular
- Replaced a lot of Razor code
- Learned to work with…
  - Modules and controllers
  - $http data service
  - Data binding
- Eliminated all post backs



Course Summary

- Built drop down lists
- Built HTML tables
- Worked with UI state
- Added, edited and deleted data
- Learned to validate data

- I hope you enjoyed this course!