Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS | Austin

Angular Forms and Validation

Deborah Kurata
Consultant | Developer | Mentor

Level: Intermediate

AUSTIN Code Trip NAVIGATE THE .NET HIGHWAY

Deborah Kurata

- Independent Consultant | Developer | Mentor
  - Web (Angular), .NET
- Pluralsight Author
  - AngularJS Line of Business Applications
  - Angular Front to Back with Web API
  - Object-Oriented Programming Fundamentals in C#
- Microsoft MVP

Session Materials & Sample Code

https://github.com/DeborahK/VSLive2015-Austin

Rate Yourself on Angular Forms

- New to Angular - no proficiency
- Just starting out - limited proficiency
- Doing it, not fully understanding it - working proficiency
- Been there, done that, can help others - full proficiency

Overview

- Building a data entry form
- Basic validation
- Validation messages
- Data Binding Options
- Custom validators
- Asynchronous validators
- Peek at Angular 2 Forms

Demo: Angular in Action

```
<form name="movieForm">
  <fieldset>
    <legend>Edit Movie Information</legend>
  </fieldset>
</form>
```

Creating a Form

It's just HTML

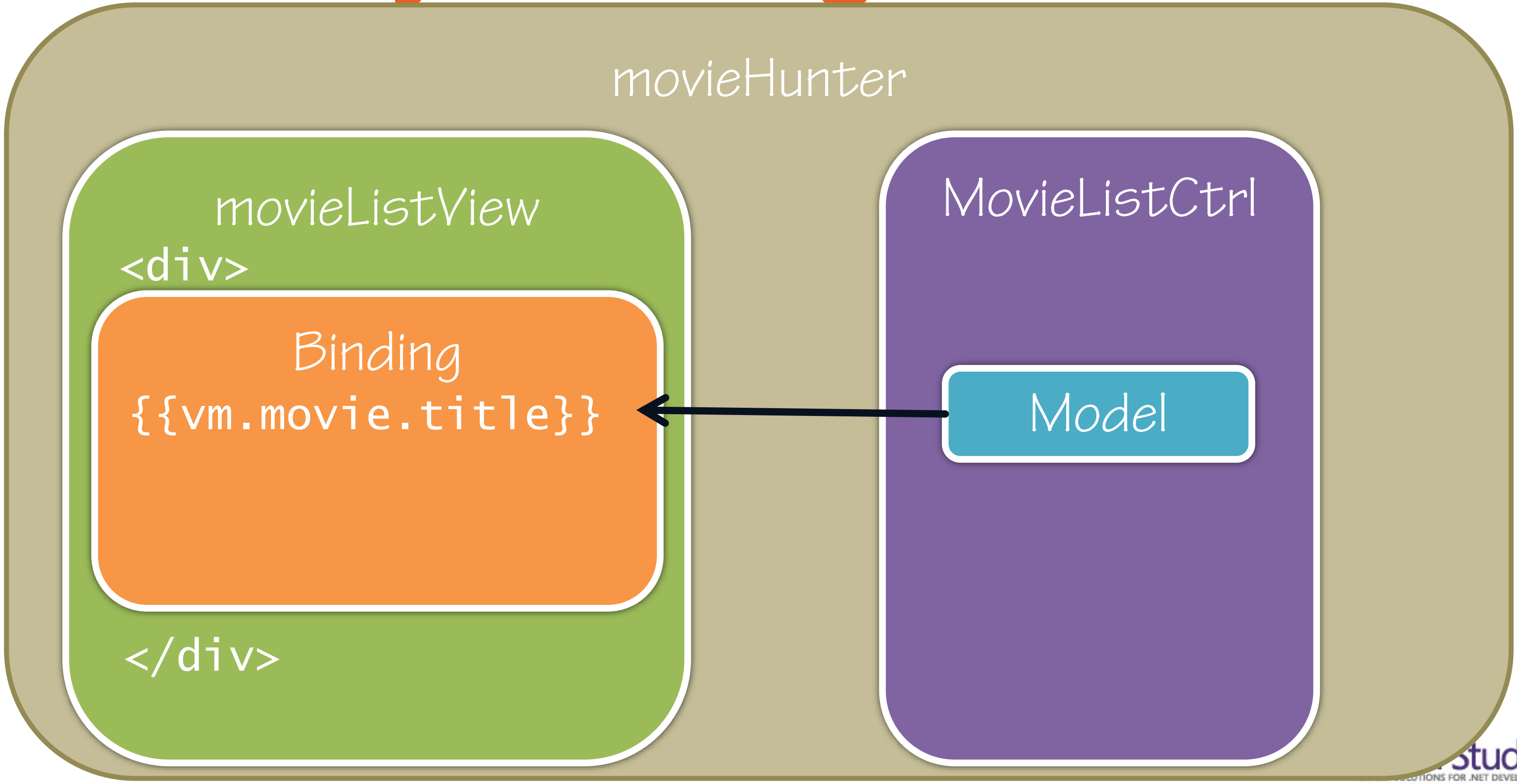
Adding Controls to a Form

```
<div>
  <label for="inputMovieTitle">
         Movie Title</label>
  <input id ="inputMovieTitle“
         type="text"
         placeholder="Movie Title" />
</div>
<div>
  <label for="inputDescription">
         Description</label>
  <textarea id="inputDescription“
            placeholder="Description"
            rows="3" />
</div>
```

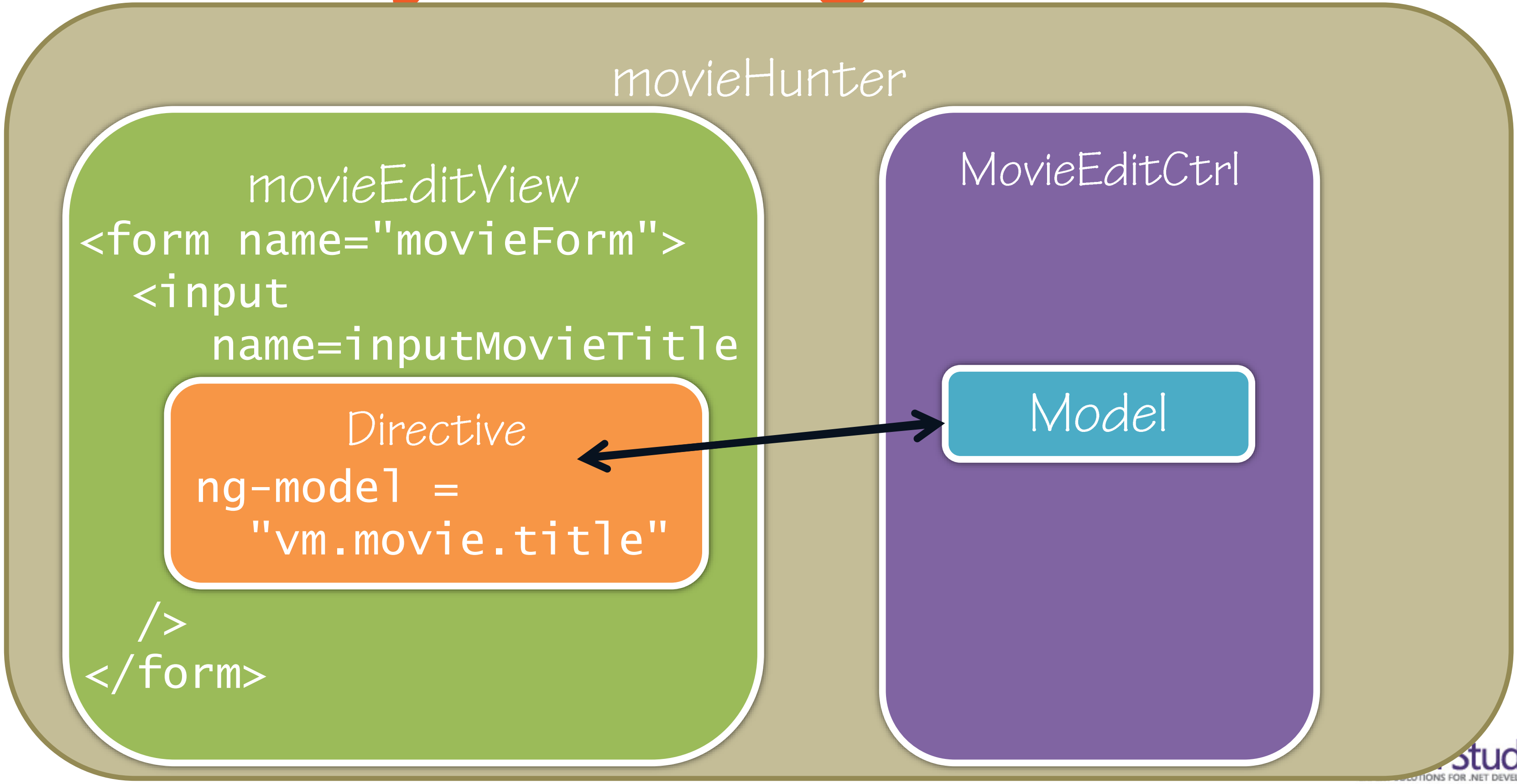
Data Binding

- One-Way Binding
  - View is the projection of the model
  - When the model changes, the view reflects the change
- Two-Way Binding
  - View and model data are synchronized
  - Changes to the model are reflected in the view
  - Changes in the view are reflected in the model

One-Way Binding



Two-Way Binding



```
<form name="movieForm">
  <input
     name=inputMovieTitle
    ng-model =
      "vm.movie.title"
  />
</form>
```

Two-Way Binding

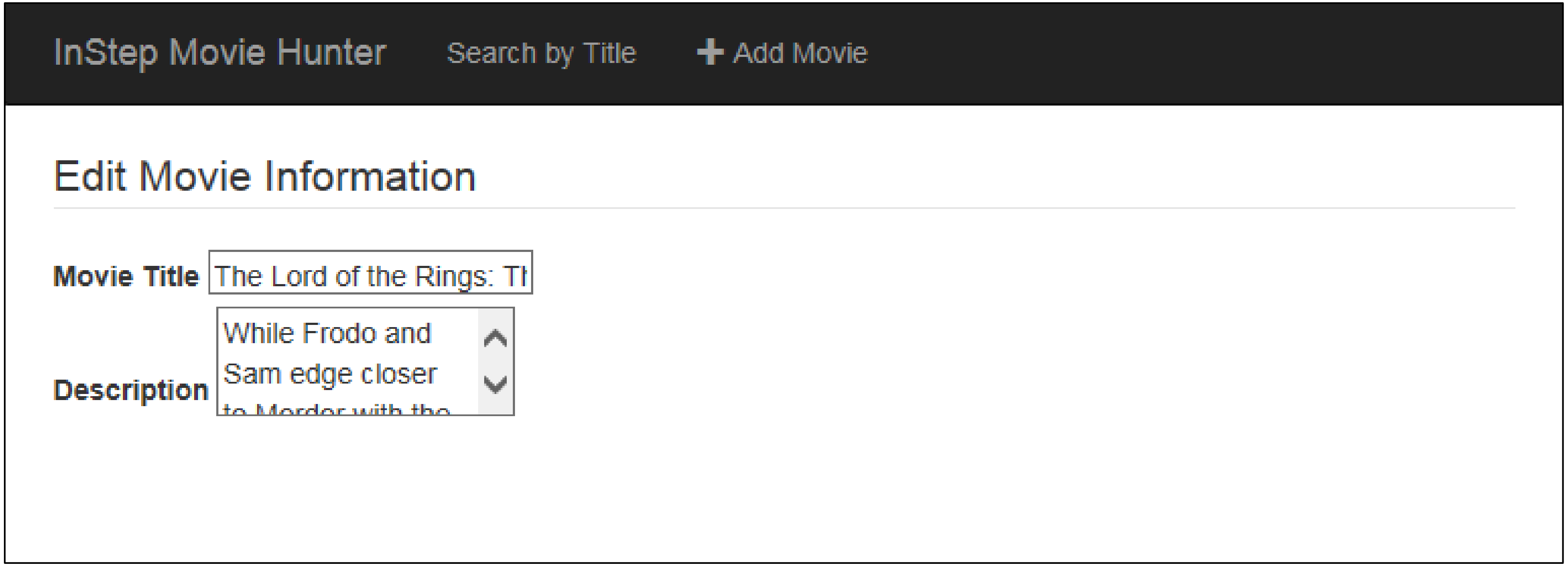
```
<div>
  <label for="inputMovieTitle">
         Movie Title</label>
  <input id ="inputMovieTitle“
         type="text"
         placeholder="Movie Title" />
</div>
<div>
  <label for="inputDescription">
         Description</label>
  <textarea id="inputDescription“
            placeholder="Description"
            rows="3" />
</div>
```

Two-Way Binding

```
<div>
  <label for="inputMovieTitle">
         Movie Title</label>
  <input id ="inputMovieTitle“
         type="text"
         placeholder="Movie Title"
         ng-model="vm.movie.title" />
</div>
<div>
  <label for="inputDescription">
         Description</label>
  <textarea id="inputDescription“
            placeholder="Description"
            rows="3"
            ng-model="vm.movie.description" />
</div>
```

Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS

The Result



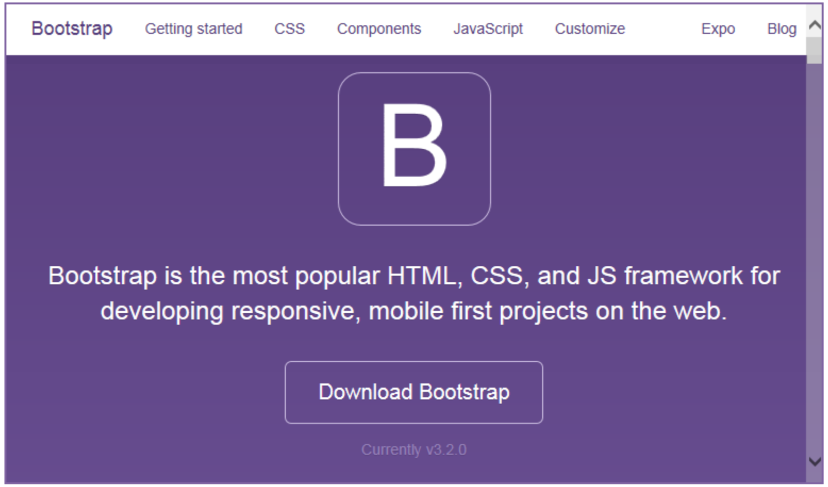
Edit Movie Information

Movie Title The Lord of the Rings: Th

Description While Frodo and Sam edge closer to Mordor with the

Bootstrap

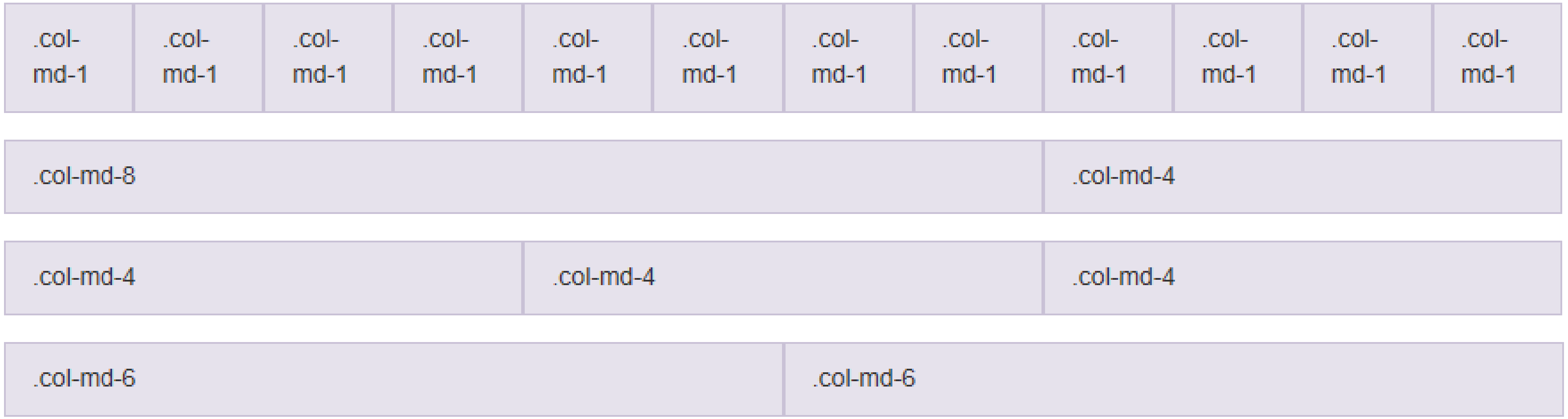
- Framework for prettifying Web pages
- Developed by Twitter
  - http://getbootstrap.com/
- Large third party community



Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

Visual Studio LIVE!
EXPERT SOLUTIONS FOR .NET DEVELOPERS

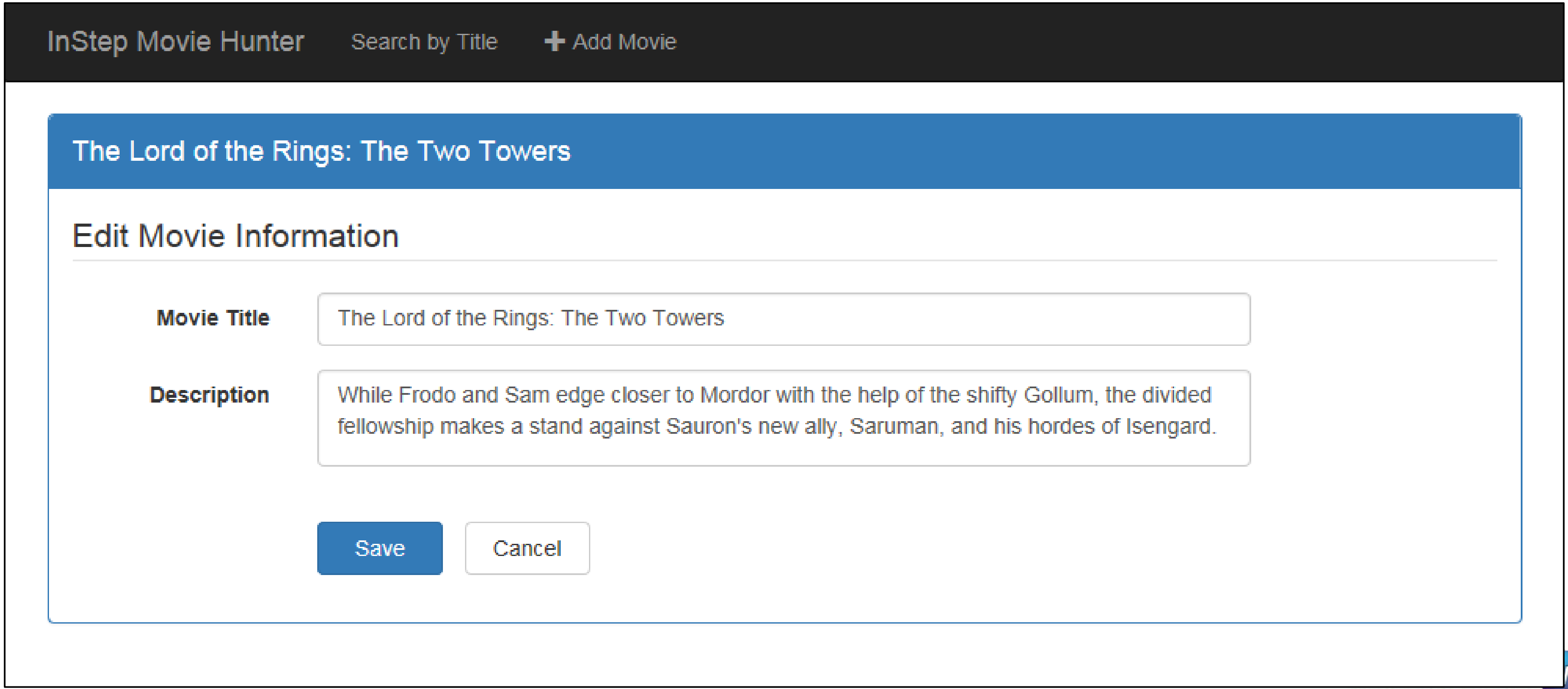
Bootstrap Grid System



Bootstrap Styles

```
<div class="form-group">
  <label class="col-md-2 control-label"
         for="inputMovieTitle">
         Movie Title</label>
  <div class="col-md-6">
        <input class="form-control"
         id ="inputMovieTitle"
         type="text"
         placeholder="Movie Title"
         ng-model="vm.movie.title" />
  </div>
</div>
```

The Result

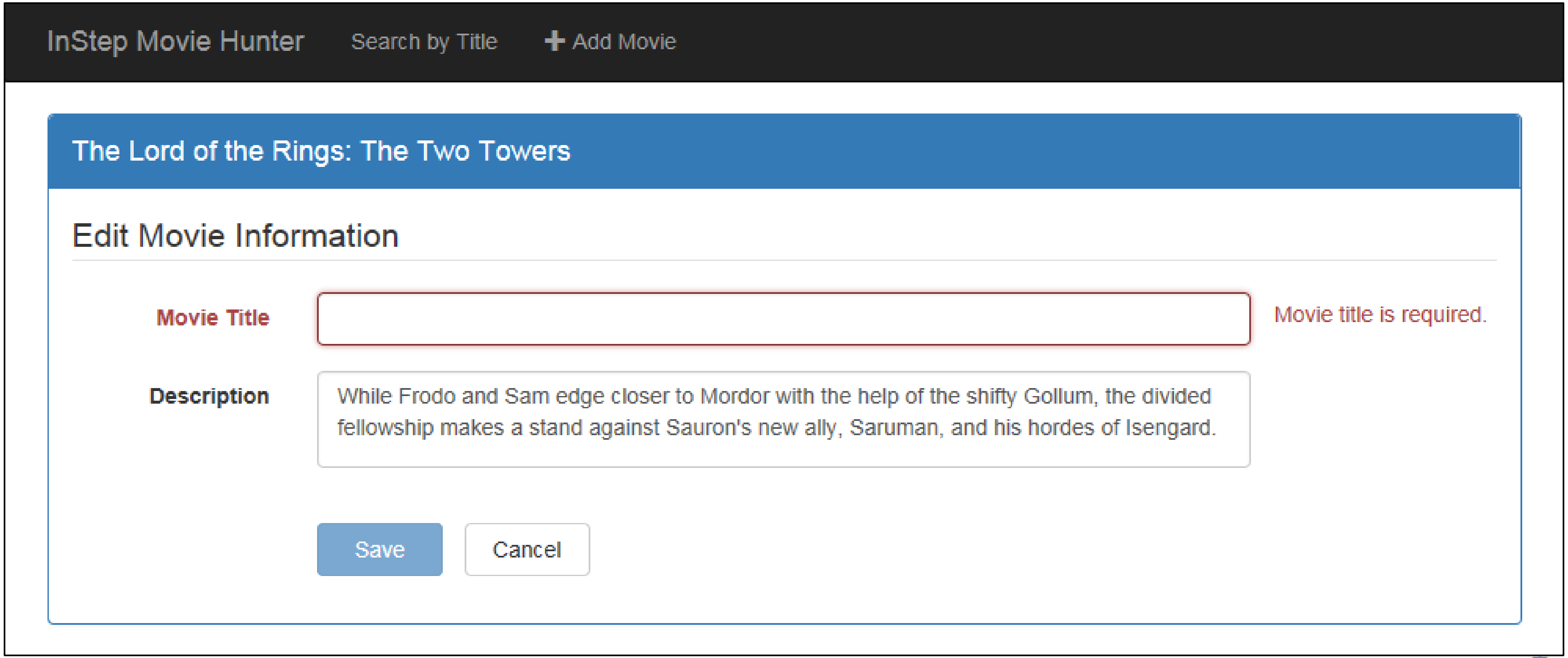


The Lord of the Rings: The Two Towers

Edit Movie Information

Movie Title	The Lord of the Rings: The Two Towers
Description	While Frodo and Sam edge closer to Mordor with the help of the shifty Gollum, the divided fellowship makes a stand against Sauron's new ally, Saruman, and his hordes of Isengard.

Validation



The Lord of the Rings: The Two Towers

Edit Movie Information

Movie Title

Movie title is required.

Description

While Frodo and Sam edge closer to Mordor with the help of the shifty Gollum, the divided fellowship makes a stand against Sauron's new ally, Saruman, and his hordes of Isengard.

Prepare the form

Ensure the form has a name

Set the novalidate attribute

```
<form class="form-horizontal"
      name="movieForm">
      novalidate>
```

Validation Techniques

Type Attribute	`type="email"`
Required Attribute	`required`
HTML Validation Attributes	`min = "1"`
Angular Validation Attributes	`ng-minlength = "3"`
Custom Angular Directive	`my-directive=""`

Validation Attributes

```
<div class="col-md-8">
    <input class="form-control"
            id="inputMovieTitle"
            name="inputMovieTitle"
            type="text"
            placeholder="Title (required)"
            ng-model="vm.movie.title"
            required
            ng-minlength="3"
            ng-maxlength="50" />
</div>
```

Marking Invalid Elements

ngClass Directive

Bootstrap Style

Angular Validation States

```
<div class="form-group"
     ng-class="{'has-error':
                movieForm.inputMovieTitle.$invalid &&
                movieForm.inputMovieTitle.$dirty}">
</div>
```

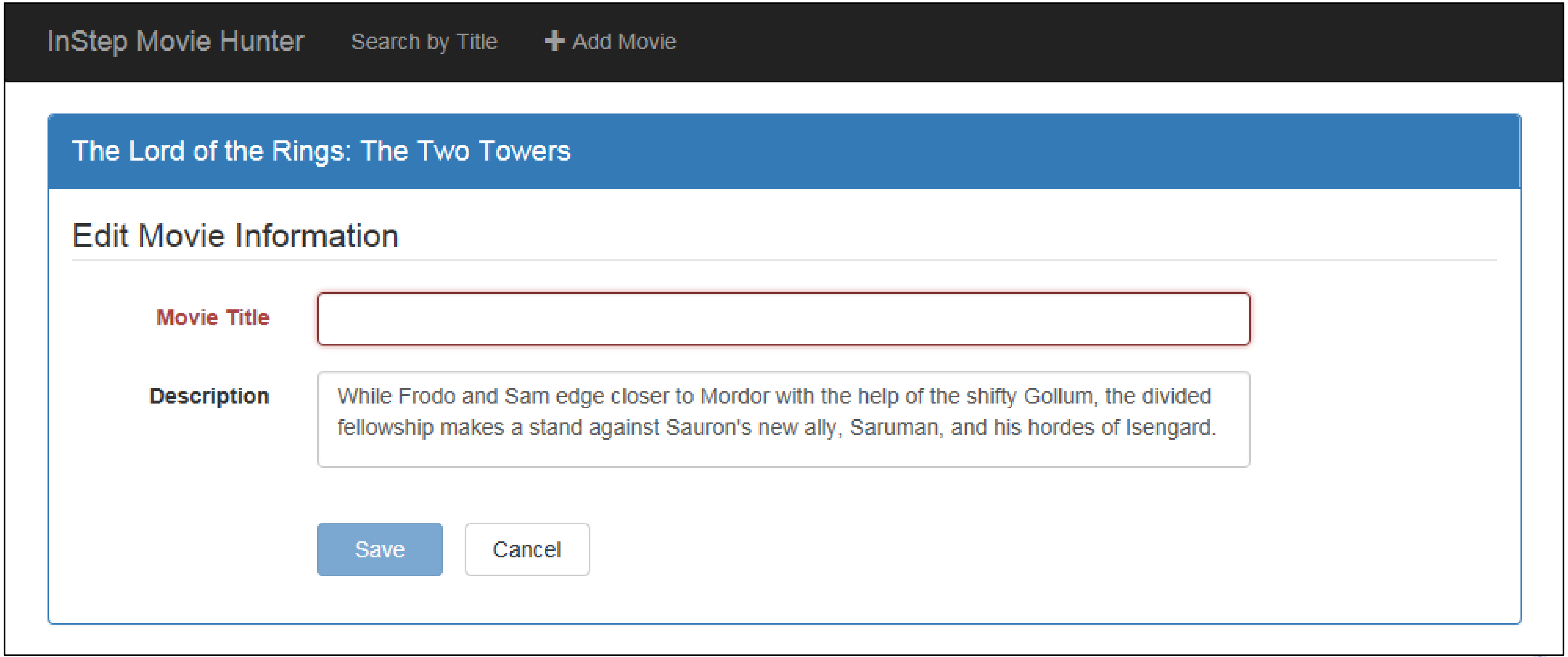
Angular Validation States

$pristine	Value has not been changed
$dirty	Value has been changed
$valid	Value is valid
$invalid	Value is invalid
$error	All validations and whether they are valid or invalid

Validation States: New in 1.3

$touched	Control was "touched" (on blur)
$untouched	Control never "touched" (blur)

Validation Messages



The Lord of the Rings: The Two Towers

Edit Movie Information

Movie Title

Description: While Frodo and Sam edge closer to Mordor with the help of the shifty Gollum, the divided fellowship makes a stand against Sauron's new ally, Saruman, and his hordes of Isengard.

ngMessages: New in 1.3

Requires

- angular-messages
- ngMessages

ng-Messages

- Shows or hides messages
- Similar to a switch or case statement

ng-Message

- Defines specific cases

```
<span class="help-block">
  <span ng-messages="movieForm.inputMovieTitle.$error">
    <span ng-message="required">Movie title is required</span>
    <span ng-message="minlength">Must be at least 3 characters</span>
    <span ng-message="maxlength">Cannot exceed 50 characters</span>
  </span>
</span>
```

Demo: Forms & Validation

Creating a Messages File

```
validationMessages.html

<span ng-message="required">Value is required</span>
<span ng-message="minlength">Value is too small</span>
<span ng-message="maxlength">Value is too long</span>
```

```
movieEditView.html

<span ng-messages="movieForm.inputMovieTitle.$error"
      ng-messages-include="app/validationMessages.html">
</span>
```

Demo: Messages File

Data Binding Options

- By default, synchronization occurs as the user types
- Use ng-model-options to change when the synchronization occurs
  - {updateOn: 'blur'}
  - {debounce: 1500}
  - {updateOn: 'submit'}

Demo: Binding Options

Rolling Back Changes to a Field

- If using ng-model-options, the model is not changed as the user types
- Changes to a field can be rolled back.
  - Use ng-model-options to change when the synchronization occurs
  - ng-keyup="vm.cancelEntry(movieForm. inputMovieTitle, $event)"
  - vm.cancelEntry = **function** (control, event) {

```
vm.cancelEntry = function (control, event) {
    if (event.keyCode == 27) {
        control.$rollbackViewValue();
    }
};
```

Demo: Rolling Back Changes

Custom Validator

- Developed as a directive
- ngModel.$validators
  - Collection of validators
  - Returns true/false
- Used as an attribute

```
directorMinRatingValidator.js

function directorMinRatingValidator() {
   . . .
}
```

```
movieEditView.html
<input  id="inputStarRating"
        type="number"
        ng-model="vm.movie.starRating"
        required
        director-min-rating-validator="vm.movie.director"
        min="1"
        max="5"/>
```

Demo: Custom Validators

Asynchronous Validator

- Developed as a directive
- ngModel.$asyncValidators
  - Collection of validators
  - Returns promise
- Used as an attribute

```
duplicateTitleValidator.js

function duplicateTitleValidator () {
   . . .
}
```

```
movieEditView.html
<input  id="inputMovieTitle"
        type="text"
        ng-model="vm.movie.title"
        required
        duplicate-title-validator="vm.movie.movieId"
        ng-minlength="3"
        ng-maxlength="50"/>
```

Demo: Async Validators

Angular 2 Forms

https://angular.io/

Angular 2 Forms

- Will be an external module
  - Like ngMessage and ngResource
- Three techniques

Imperative Driven	Data Driven	Template Driven
Built w/Code	Automatic	Similar to 1.x

Angular 2: Component Definition

Annotations

```
@Component({
   selector: 'todo-app'
})
@View({
   url: 'todos.html'
})
```

Controller

```
Class TodoApp {
  todos: [string];
  constructor() {
     this.todos = ['Item1', 'Item2'];
  }
  addTodo(title: string) {
    this.todos.push(title);
  }
}
```

Form Element

```
<todo-app></todo-app>
```

Angular 2: View

Local Variable #localvar

Event Handler (event)

Property Bindings [innerText]

```
<input
        type="text"
        #toDoTextbox />



<button (click)="addTodo(toDoTextbox.value)">Add</button>


<div [innertext]="toDoTextbox.value"></div>
```

Controls & Validators

```
var builder = new FormBuilder();

// create a control group
var group = builder.group({
   firstName: ["Chris", Validators.required],
   lastName: ["Smith", Validators.required],
   address: [""]
});


// check states
Var isValid = group.valid;
Var isChanged = group.dirty;
Var numErrors = group.errors.length;
```

Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS

Example: Component

```
import {bootstrap, Component, View} from 'angular2/angular2';
import {FormBuilder, Validators, FormDirectives, ControlGroup} from 'angular2/forms';

@Component({ selector: 'form-app', injectables: [FormBuilder]})
@View({ url: 'form-app.html', directives: [FormDirectives]})

class FormApp {
   infoForm: ControlGroup;

   constructor(builder: FormBuilder) {
     this.infoForm = builder.group({
        firstName: ["", Validators.required],
        lastName: ["", Validators.required],
        address: [""]
     });
   }
}
```

Example: View

Property Binding

Control Decorator

```
<div [control-group]="infoForm">
  <input [control]="infoForm.controls.firstName">
  <input [control]="infoForm.controls.lastName">
  <input [control]="infoForm.controls.address">


  <!-- control decorator short cut -->
  <input control="firstName">
  <input control="lastName">
  <input control="address">

  {{ infoForm.controls.firstName.value }}
</div>
```

Take Away

Use client-side validation where possible

Display friendly and helpful messages

Use directives to create custom & async validation

Keep an eye on Angular 2 progress

Thank You!

- @deborahkurata
- deborahk@insteptech.com
- http://blogs.msmvps.com/deborahk
- https://github.com/DeborahK/VSLive2015-Austin