The Learning House

427 S 4th Street #300

Louisville KY 40202

Introduction to jQuery Validate

.NET Cohort

Coding Bootcamp

SOFTWARE GUILD

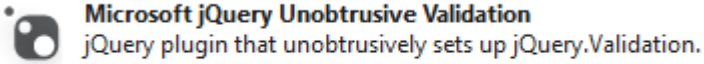
Why Do We Care?

- It is generally a break in the user experience to submit a form all the way to the server (called a “postback” or “round trip”) and have it come back with user errors.
- We prefer to catch simple validation errors on the client before bothering the server.
- We also prefer validation to clear up in real time, rather than making the user guess.

jQuery Validate Unobtrusive

Microsoft provides a NuGet plug-in called jquery.validate.unobtrusive.js. This JavaScript library works with the built-in ValidationSummary and ValidationMessageFor HTML helpers.

Note that jQuery must be installed for this to work.



Party Registration Model

```
public class RsvpResponse
{
    [Required(ErrorMessage="Please enter your name")]
    public string Name { get; set; }

    [Required(ErrorMessage = "Please enter your email")]
    [RegularExpression(@"^\S+@\S+$",
        ErrorMessage="The email address format isn't valid")]
    public string Email { get; set; }

    [Required(ErrorMessage = "Please enter your phone number")]
    public string Phone { get; set; }

    [Required(ErrorMessage = "Please enter your favorite game")]
    public string FavoriteGame { get; set; }

    [Required(ErrorMessage = "Please specify whether you will attend")]
    public bool? WillAttend { get; set; }
}
```

Next, Create a Shared Layout

```
<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <meta name="viewport" content="width=device-width" />
    <link href="~/Content/bootstrap.min.css" rel="stylesheet"/>
    <link href="~/Content/bootstrap-theme.min.css" rel="stylesheet" />
    <link href="~/Content/Styles.css" rel="stylesheet"/>
    <title>@ViewBag.Title</title>
</head>
<body>
    <div class="container">
        @RenderBody()
    </div>
    <script src="~/Scripts/jquery-1.9.1.min.js"></script>
    <script src="~/Scripts/bootstrap.min.js"></script>
    @RenderSection("Scripts", false)
</body>
</html>
```

Don't forget jquery!

SOFTWARE GUILD

@RenderSection

- With more scripts into our web application, we might realize that we are referencing a lot of scripts that aren't needed on every page.
- MVC allows us to define sections in our layout pages and declare whether they are optional or not.
- For our extra JavaScript files, let's render an optional section called scripts at the bottom of our layout page.
- If false is passed, it is optional. If true is passed, it will error out if the view doesn't define a Scripts section.

```
@RenderSection("Scripts", false)
```

Create the Controller

```
// GET: /Home/RegistrationForm
public ActionResult RegistrationForm()
{
    var model = new RsvpResponse();

    return View(model);
}

// POST: /Home/RegistrationForm
[HttpPost]
public ActionResult RegistrationForm(RsvpResponse model)
{
    if (ModelState.IsValid)
        return View("Thanks", model);
    else
        return View(model);
}
```

Create the First View

```
<div class="row">
    <div class="col-xs-6">
        <h3>RSVP Form</h3>
        @using (Html.BeginForm("RegistrationForm", "Home", null, FormMethod.Post, new { id = "rsvpForm" }))
        {
            @Html.ValidationSummary()

            <div class="form-group">
                <label>Your Name:</label>
                @Html.TextBoxFor(m => m.Name, new { @class = "form-control" })
            </div>
            <div class="form-group">
                <label>Your Email: </label>
                @Html.TextBoxFor(m => m.Email, new { @class = "form-control" })
            </div>
            <div class="form-group">
                <label>Your Phone:</label>
                @Html.TextBoxFor(m => m.Phone, new { @class = "form-control" })
            </div>
            <div class="form-group">
                <label>Your Favorite Game:</label>
                @Html.TextBoxFor(m => m.FavoriteGame, new { @class = "form-control" })
            </div>
            <div class="form-group">
                <label>Will you attend?</label>
                @Html.DropDownListFor(m => m.WillAttend,
                                      new[] {
                                                new SelectListItem { Text = "Yes", Value = bool.TrueString},
                                                new SelectListItem { Text = "No", Value = bool.FalseString},
                                            },
                                  "Choose an option", new { @class = "form-control" })

            </div>
            <input type="submit" value="Submit RSVP" class="btn btn-primary" />
        }
    </div>
</div>
```

Add the Scripts section

Link in the jquery.validate.unobtrusive script…

```
            </div>
            <input type="submit" value="Submit RSVP" class="btn btn-primary" />
        }
    </div>
</div>

@section Scripts
{
    <script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>
}
```

SOFTWARE GUILD

Create the Confirmation View

```
@model MVC_RSVP.Models.RsvpResponse

@{
    Layout = "~/Views/Shared/_Layout.cshtml";
    ViewBag.Title = "Thanks";
}

<div class="row">
    <div class="col-xs-12">
        <h1>Thank you, @Model.Name</h1>
        @if (Model.WillAttend.Value)
        {
            <p>
                It's great that you're coming, we will see if others
                like @Model.FavoriteGame and play it if so!
            </p>
        }
        else
        {
            <p>Sorry you couldn't make it this time</p>
        }
    </div>
</div>
```

Make Sure Unobtrusive Validation is Enabled

The Web.config appSettings has a boolean key pair for this.

```
<configuration>
  <appSettings>
    <add key="webpages:Version" value="2.0.0.0" />
    <add key="webpages:Enabled" value="false" />
    <add key="PreserveLoginUrl" value="true" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  </appSettings>
```

Give it a Try!

Notice: if you put a break point in the post, it does not do the postback if the form is invalid.

The jQuery unobtrusive JavaScript checked the form against the model and prevented submission when validation failed.

How?

If you view the page source, you will notice that there are data-val attributes on the inputs. The JavaScript provided by Microsoft uses these to validate and push messages to the validation summary.

It blocks form submission if any fields fail to validate.

```
<input class="form-control" data-val="true" data-val-required="Please enter your name"
       id="Name" name="Name" type="text" value="" />

<input class="form-control" data-val="true" data-val-regex="The email address format isn&#39;t valid"
       data-val-regex-pattern="^\S+@\S+$" data-val-required="Please enter your email"
       id="Email" name="Email" type="text" value="" />
```

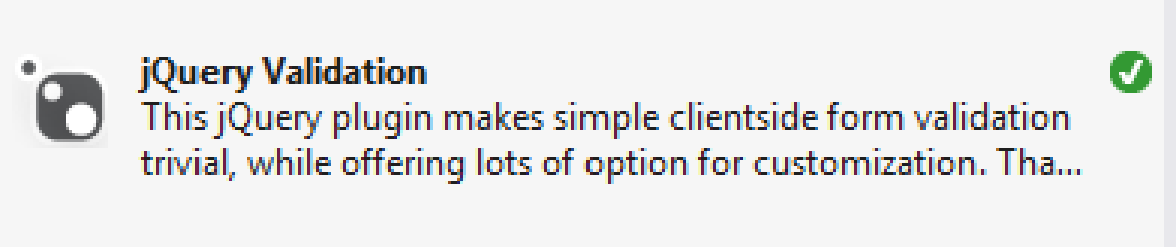
This is nice and all…

But Bootstrap has some awesome classes like has-error, has-success, etc. for highlighting forms and generally making things look nice.

Can we use that instead?

Enter jQuery Validation

- jQuery Validation is a popular validation plug-in for jQuery.
- jQuery must be on the page you will use it for.
- Where do we get jQuery Validation?
  Duh, NuGet!



Let’s make a new action

Let’s copy the RegistrationForm view over to a new view.

Creatively, we will call this “RegistrationForm2.”

RegistrationForm2

Remove the ValidationSummary and give this form an id. (Validation binds itself to forms via the id.)

```
@using (Html.BeginForm("RegistrationForm", "Home", null, FormMethod.Post, new { id = "rsvpForm" }))
{
    <div class="form-group">
        <label>Your Name:</label>
        @Html.TextBoxFor(m => m.Name, new { @class = "form-control" })
```

Create a JavaScript File

rsvpValidation.js in the Scripts/app folder.

(I like to create a folder in Scripts called /app where I put my custom JavaScript, to keep it separate from third-party downloads.)

rsvpValidation.js

```
$(document).ready(function() {
    $('#rsvpForm').validate({
        rules: {
            Name: {
                required: true
            },
            Email: {
                required: true,
                email: true
            },
            Phone: {
                required: true
            },
            FavoriteGame: {
                required: true
            },
            WillAttend: {
                required: true
            }
        }
    });
});
```

When the document is ready… locate the form with id rsvpForm and validate it with the following rules:

- The input with id Name is required.
- The input with id Email is required and must be an email address format.
- The inputs for Phone, FavoriteGame, and WillAttend are required.

Built-in Rules

Rule	Description	Usage
required	Make the field required	required: true
minlength	Require a minimum length	minlength: 3
maxlength	Require a maximum length	maxlength: 10
rangelength	Require a range value	rangelength: [3, 10]
min	Set a minimum value	min: 13
max	Set a maximum value	max: 100
range	Set a value range	range: [75, 100]
email	Value must be email format	email: true
url	Value must be URL format	url: true
date	Value must be date format	date: true
number	Value must be numeric	number: true
digits	Value must be an int	digits: true
creditcard	Credit card Format	creditcard: true

And More!

There are additional plug-ins and add-ons for jQuery Validation provided by the developer community. If you need something special, just dig around a bit.

A list of all rules and features can be found here:

http://jqueryvalidation.org/documentation/

But Surely…

jQuery Validation doesn't play nice with Bootstrap out of the box?

Right. It's a general purpose framework so, by default, it will just highlight fields. Lucky for us, it has customization options so we can instruct it on how to behave with our Bootstrap form.

jquery-validate-defaults.js

```
$.validator.setDefaults({
    highlight: function (element) {
        $(element).closest('.form-group').removeClass('has-success').addClass('has-error');
    },
    unhighlight: function (element) {
        $(element).closest('.form-group').removeClass('has-error').addClass('has-success');
    },
    errorElement: 'span',
    errorClass: 'help-block',
    errorPlacement: function (error, element) {
        if (element.parent('.input-group').length) {
            error.insertAfter(element.parent());
        } else {
            error.insertAfter(element);
        }
    }
});
```

Whut?

Default Property	Description
element	The input field we are validating
highlight	Function to run when validation fails
unhighlight	Function to run when validation errors are fixed
errorElement	The HTML element to put the validation message into
errorClass	The CSS class to put on the errorElement
errorPlacement	Where to put the errorElement physically on the page

Highlight

```
highlight: function (element) {
    $(element).closest('.form-group').removeClass('has-success').addClass('has-error');
},
```

“Find the closest .form-group, then remove the has-success class, if present, and then add the has-error class.”

Un-Highlight

```
unhighlight: function (element) {
    $(element).closest('.form-group').removeClass('has-error').addClass('has-success');
},
```

"Find the closest .form-group, then remove the has-error class, if present, and then add the has-success class."

error*

Create a span with class help-block. If the element we are validating is in an input-group, put it at the end of the whole group. Otherwise, put it after this element.

This field is required

```
errorElement: 'span',
errorClass: 'help-block',
errorPlacement: function (error, element) {
    if (element.parent('.input-group').length) {
        error.insertAfter(element.parent());
    } else {
        error.insertAfter(element);
    }
}
```

Let’s try it out!

Add the jquery.validate main script, our defaults file, and our custom validation file to the Scripts section of RegistrationForm2.

The order is important!

```
@section Scripts
{
    <script src="~/Scripts/jquery.validate.min.js"></script>
    <script src="~/Scripts/app/jquery-validate-defaults.js"></script>
    <script src="~/Scripts/app/rsvpValidation.js"></script>
}
```

SOFTWARE GUILD

Very Nice!

Your Name:

Eric

Your Email:

a

Please enter a valid email address.

Your Phone:

This field is required.

Your Favorite Game:

This field is required.

Will you attend?

Choose an option

This field is required.

Can I Change the Messages?

Sure! Let's add a minlength of 3 to FavoriteGame for demonstration purposes as well.

```
        FavoriteGame: {
            required: true,
            minlength: 3
        },
        WillAttend: {
            required: true
        }
    },
    messages: {
        Name: "Enter your name",
        Email: {
            required: "Enter your email address",
            email: "That's not a format for email I'm aware of..."
        },
        Phone: "Enter your phone number",
        FavoriteGame: {
            required: "Tell us your favorite game",
            minlength: $.validator.format("I don't know of any game with less than {0} characters...")
        },
        WillAttend: "We need to know if you will attend!"
    }
```

Conclusion

For quick validation, the built-in unobtrusive validation works well with the validation summary and validation message controls.

For fine-grained control and integration with other style packages, like Bootstrap, the jQuery Validation plug-in allows for quick and easy custom validation.