

# Implementing Data Validation

---



**Alex Wolf**

[www.alexwolfthoughts.com](http://www.alexwolfthoughts.com)



# To-Do List



**Validating Form Inputs in Web Forms**

**Understanding Data Validation in MVC**

**Demo: Adding Validation to the Task Form**

**Implementing Custom Validation in MVC**

**Demo: Creating Custom Data Attributes**

**Demo: Customizing Model Level Validation**



# Exploring Validation in MVC

---



# Understanding Data Attributes

```
public class Task
{
    public int Id { get; set; }

    [Required]
    public string Title { get; set; }

    [Required]
    public string Description { get; set; }

    // Other Properties
}
```



# The Default **Validation** Attributes

**Required**

**StringLength**

**RegularExpression**

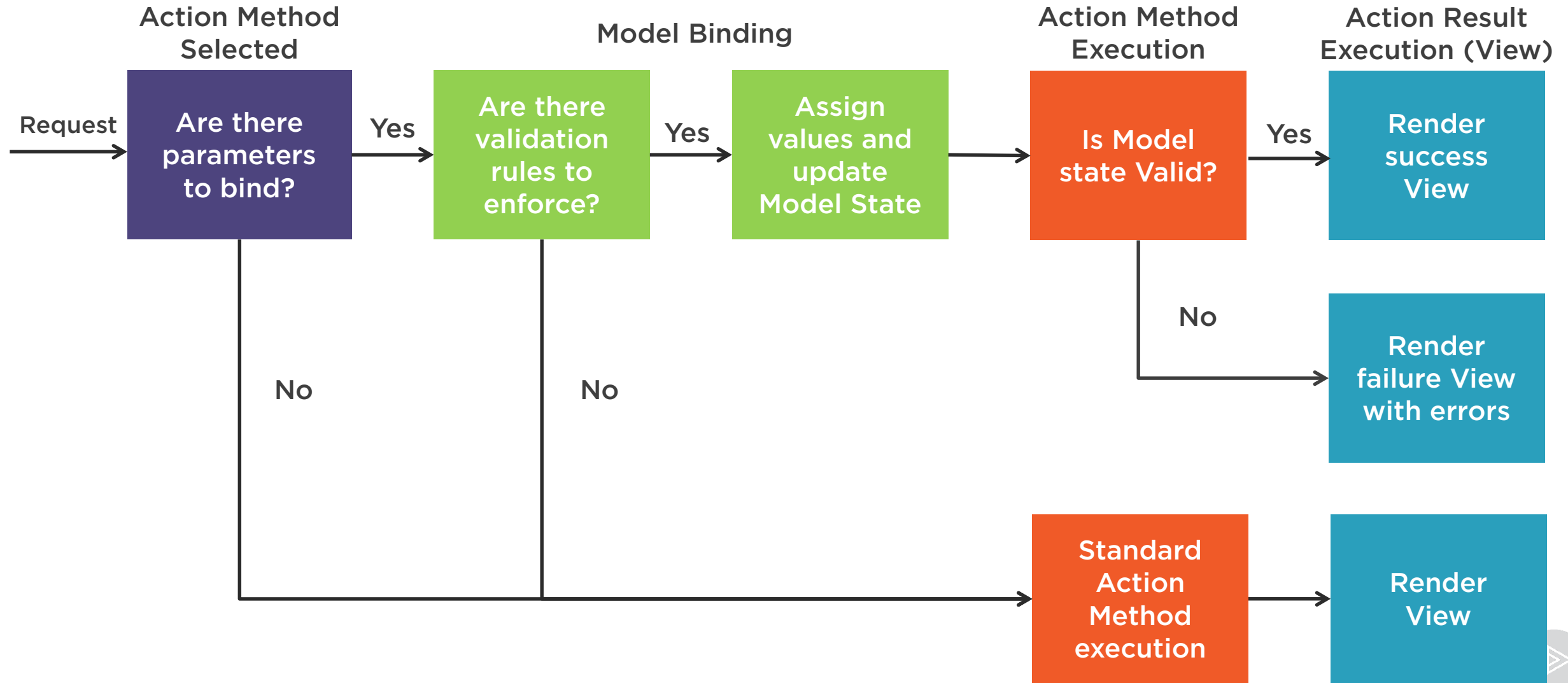
**Range**

**Compare**

**Remote**



# Data Validation and the MVC Pipeline



```
<div class="form-group">  
    @Html.LabelFor(x => x.Description, "Description")  
    @Html.TextAreaFor(x => x.Description)  
    @Html.ValidationMessageFor(x => x.Description)  
</div>
```

## Razor **Validation** Helpers

***ValidationMessageFor*** renders Property level errors



```
@Html.ValidationSummary()  
  
<div class="form-group">  
    @Html.LabelFor(x => x.Description, "Description")  
    @Html.TextAreaFor(x => x.Description)  
  
</div>
```

## Razor **Validation** Helpers (cont.)

***ValidationSummary*** renders Model and (optionally) Property level errors





```
<div class="form-group">  
    <input class="form-control" data-val="true" data-val-required="The Title field is required."  
id="Title" name="Title" type="text" value="">  
</div>
```

## Client Side Validation

**Razor Validation Helpers can auto generate data attributes for validation scripts**



# Applying Validation in MVC

---

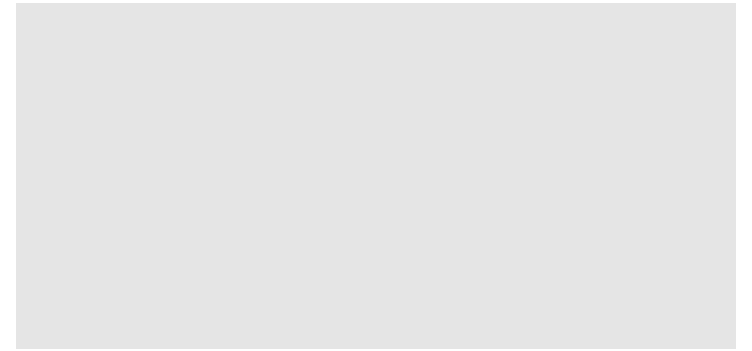
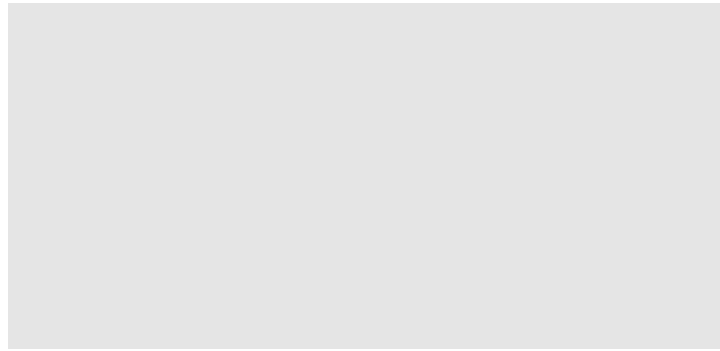
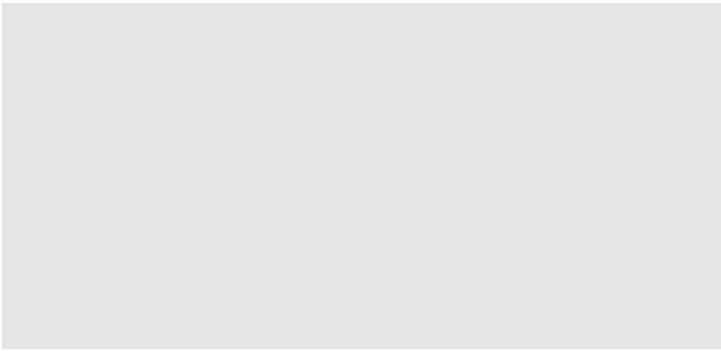


# Customizing Validation in MVC

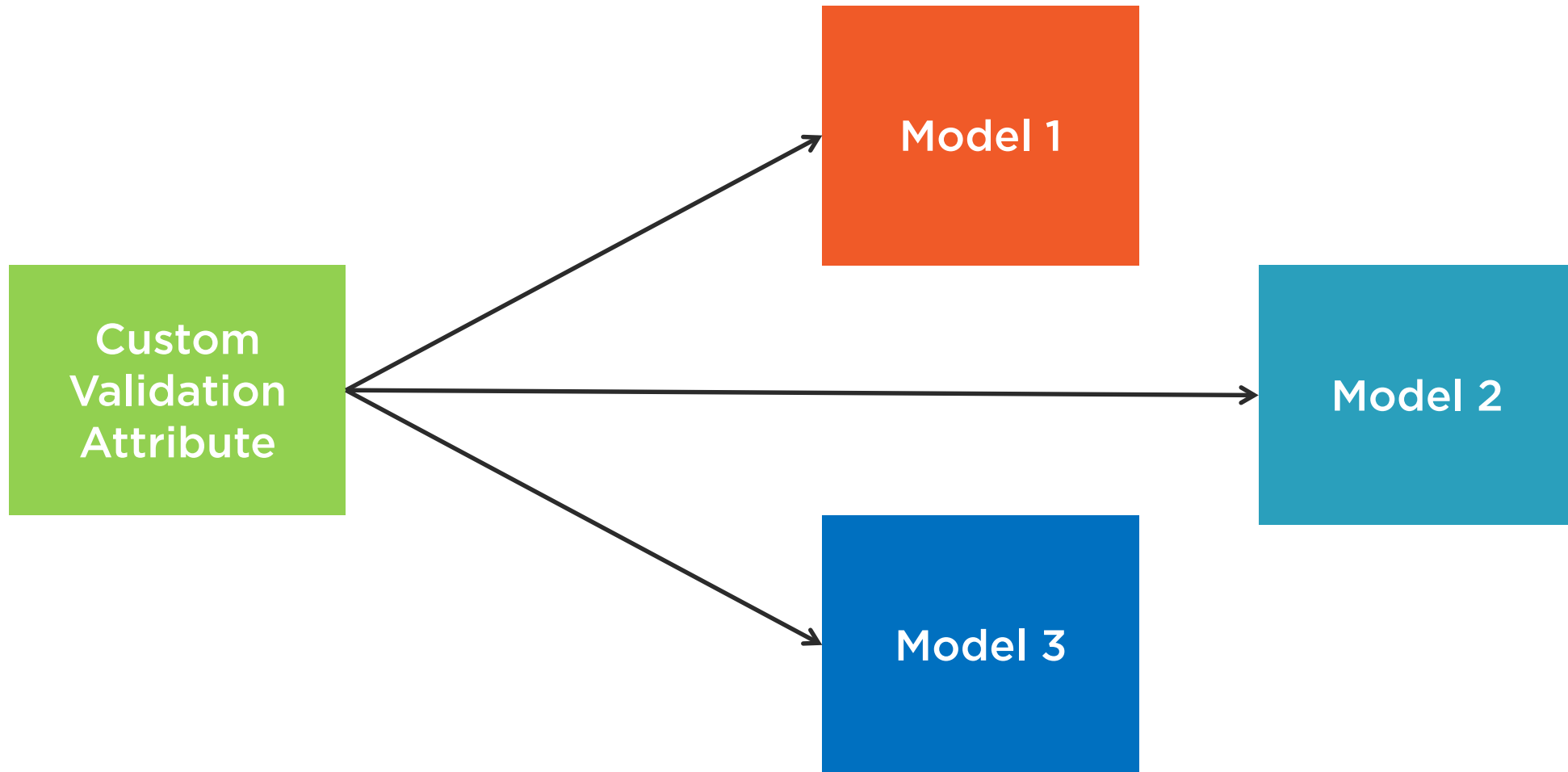
---



# Creating Custom Data Validation Attributes



# Reusable Validation Components



```
public interface IValidatableObject
{
    IEnumerable<ValidationResult> Validate(ValidationContext validationContext)
}
```

The **IValidatableObject** Interface  
Implemented on the Model rather than a specific Property



# Understanding Model Level Validation


```
public class Task
{
    public int Id { get; set; }

    [Required]
    public string Title { get; set; }

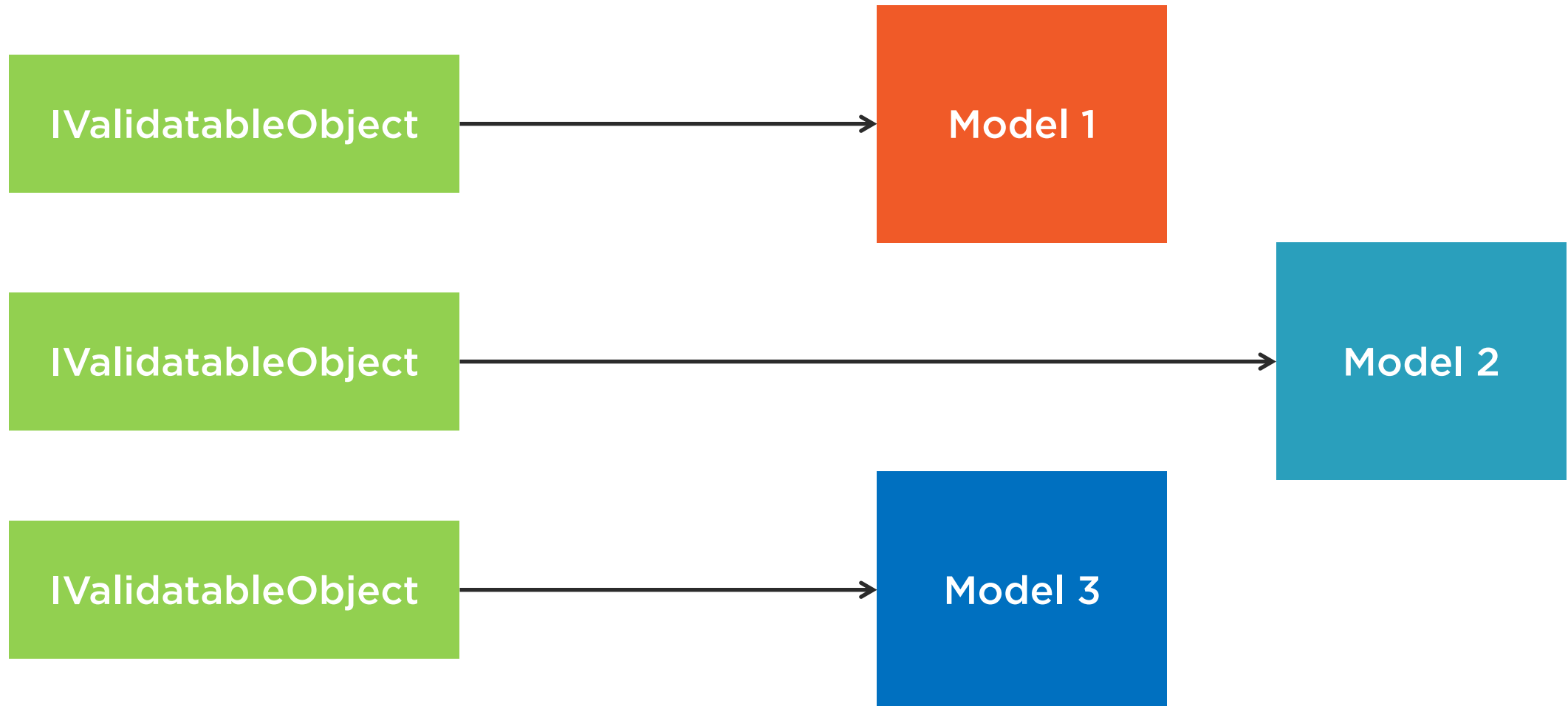
    [Required]
    public string Description { get; set; }

    public string Notes { get; set; }

    public string Completed { get; set; }
}
```



# Limited Code Reusability





```
public class Task : IValidatableObject
{
    public int Id { get; set; }

    [Required]
    public string Title { get; set; }

    [Required]
    public string Notes { get; set; }

    public IEnumerable<ValidationResult>
    Validate(ValidationContext validationContext)
    {
        //Validation Logic
    }
}
```

◀ Often not ideal to include logic in models



## Summary



Web Forms validates form data through associated server controls

MVC implements validation through the Model Binding process

Data Attributes or the `IDataValidatableObject` interface can be used to apply validation

Razor offers helper methods to streamline displaying error messages

Both framework implementations offer client side enhancements as well



# A Short Detour

---

