Adding Forms and Validation

Daniel Villamizar

Senior Cloud Solutions Architect - MVP

@danielvillamizara – https://www.linkedin.com/in/danielvillamizara/

Module overview

Understanding data binding

Creating a form with input components

Adding validation

Understanding Data Binding

```
<h1 class="page-title">
    Details for @FirstName @LastName
</h1>


public string FirstName { get; set; }
public string LastName { get; set; }
```

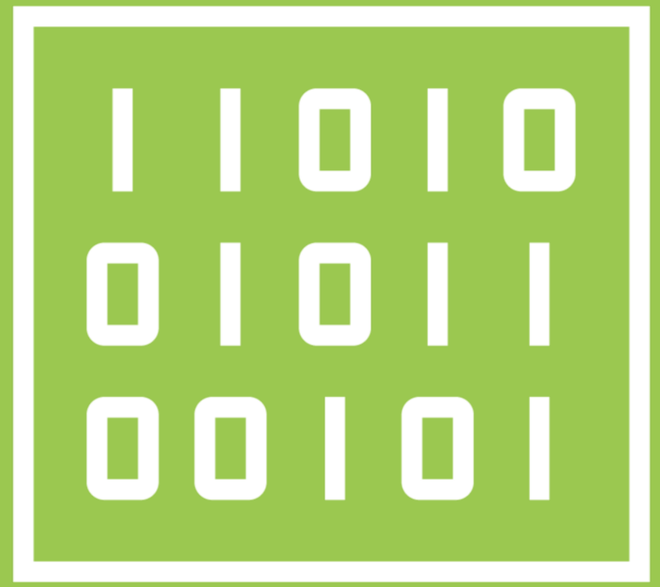
Displaying Data

One-way binding

```
<input type="text" class="form-control-plaintext">
    @Employee.FirstName
</input>

public Employee Employee { get; set; }
```

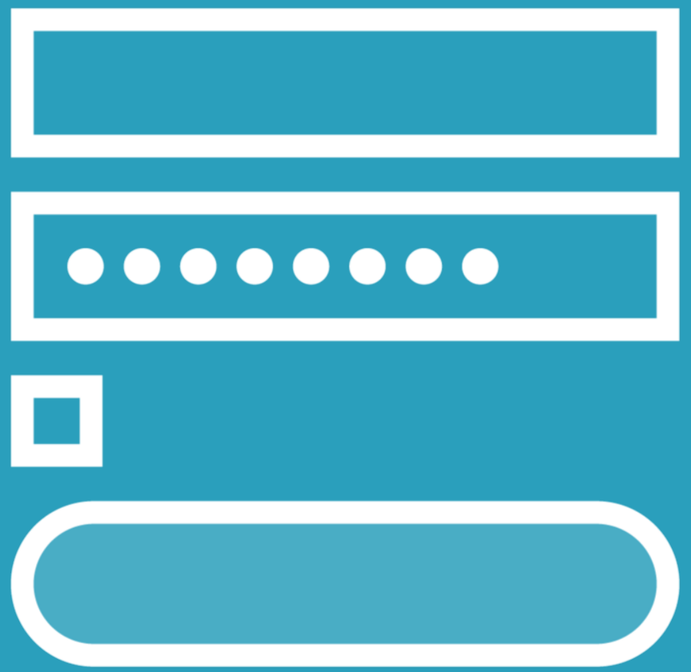
Displaying Data in a Form Control



Data Binding

@bind directive

Can bind to property, field or Razor expression



Working with a form

Allowing the user to enter data

Will typically send the data to the API

Use HTML form components or Blazor form components

```
<input id="lastName" @bind="@Employee.LastName"
     placeholder="Enter last name" />
```

Data Binding in a Form Control

Using the @bind directive
Data flows in two directions
Triggers when focus leaves the input

```
<input id="lastName" @bind-value="Employee.LastName"
    @bind-value:event="oninput"
    placeholder="Enter last name" />
```

Data Binding on a Different Event

Demo

Working with data binding

Creating a Form with Input Components

Forms in Blazor

EditForm

Built-in input components

Data binding

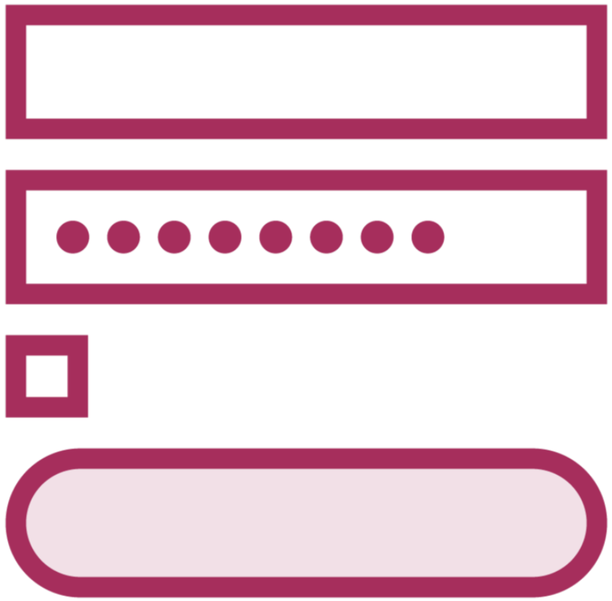
Validation support

A First EditForm

```
<EditForm Model="@Employee" >

</EditForm>
```

```
public Employee Employee { get; set; } = new Employee();
```



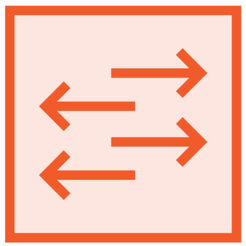
Input components

- InputText
- InputTextArea
- InputNumber
- InputSelect
- InputDate
- InputCheckbox
- InputRadio
- InputRadioGroup
- InputFile

Input Components

Inherit from InputBase

Basic validation included (depending on used component)



Support two-way data binding

Adding Input Components to the EditForm

```
<EditForm Model="@Employee">
        <InputText id="lastName"
                @bind-Value="@Employee.LastName"
                placeholder="Enter last name">
        </InputText>
</EditForm>
```

Using the InputSelect

```
<InputSelect id="country" class="form-control col-sm-8"
             @bind-Value="@Employee.CountryId">
             @foreach (var country in Countries)
             {
                          <option value="@country.CountryId">@country.Name</option>
             }
</InputSelect>
```

Submitting the Form

OnValidSubmit

OnInvalidSubmit

OnSubmit

Submitting the Form

```
<EditForm Model="@Employee"
        OnValidSubmit="@HandleValidSubmit"
        OnInvalidSubmit="@HandleInvalidSubmit">
        <InputText id="lastName"
                @bind-Value="@Employee.LastName"
                placeholder="Enter last name">
        </InputText>
        <button type="submit">Save employee</button>
</EditForm>
```

Demo

Adding the Add Employee form

Using input components

Editing an employee

Demo

Saving the data

Demo

Adding an image

Adding Validation

Validation in the EditForm

Data annotations support

DataAnnotationsValidator

ValidationSummary

```
public class Employee
{
    [Required]
    [StringLength(50, ErrorMessage = "Last name is too long.")]
    public string LastName { get; set; } = string.Empty;
}
```

Applying Data Annotations on the Model

```
<EditForm Model="@Employee"
    OnValidSubmit="@HandleValidSubmit"
    OnInvalidSubmit="@HandleInvalidSubmit">

    <DataAnnotationsValidator />
</EditForm>
```

Using the DataAnnotationsValidator

Support for validation using data annotations
Field validation on tab out of the field
Submit of form doesn’t call HandleValidSubmit

```
<EditForm Model="@Employee"
     OnValidSubmit="@HandleValidSubmit"
     OnInvalidSubmit="@HandleInvalidSubmit">

     <DataAnnotationsValidator />
     <ValidationSummary />
</EditForm>
```

Displaying Errors with the ValidationSummary

Errors are only displayed by adding this component

Demo

Applying data annotations on the model

Displaying validation errors in the form

Summary

- Forms are based on built-in components
- Input components support data binding
- Basic validation support using data annotations

Up next:
Using JavaScript from Blazor