Designing with Layouts and Views

Alex Wolf

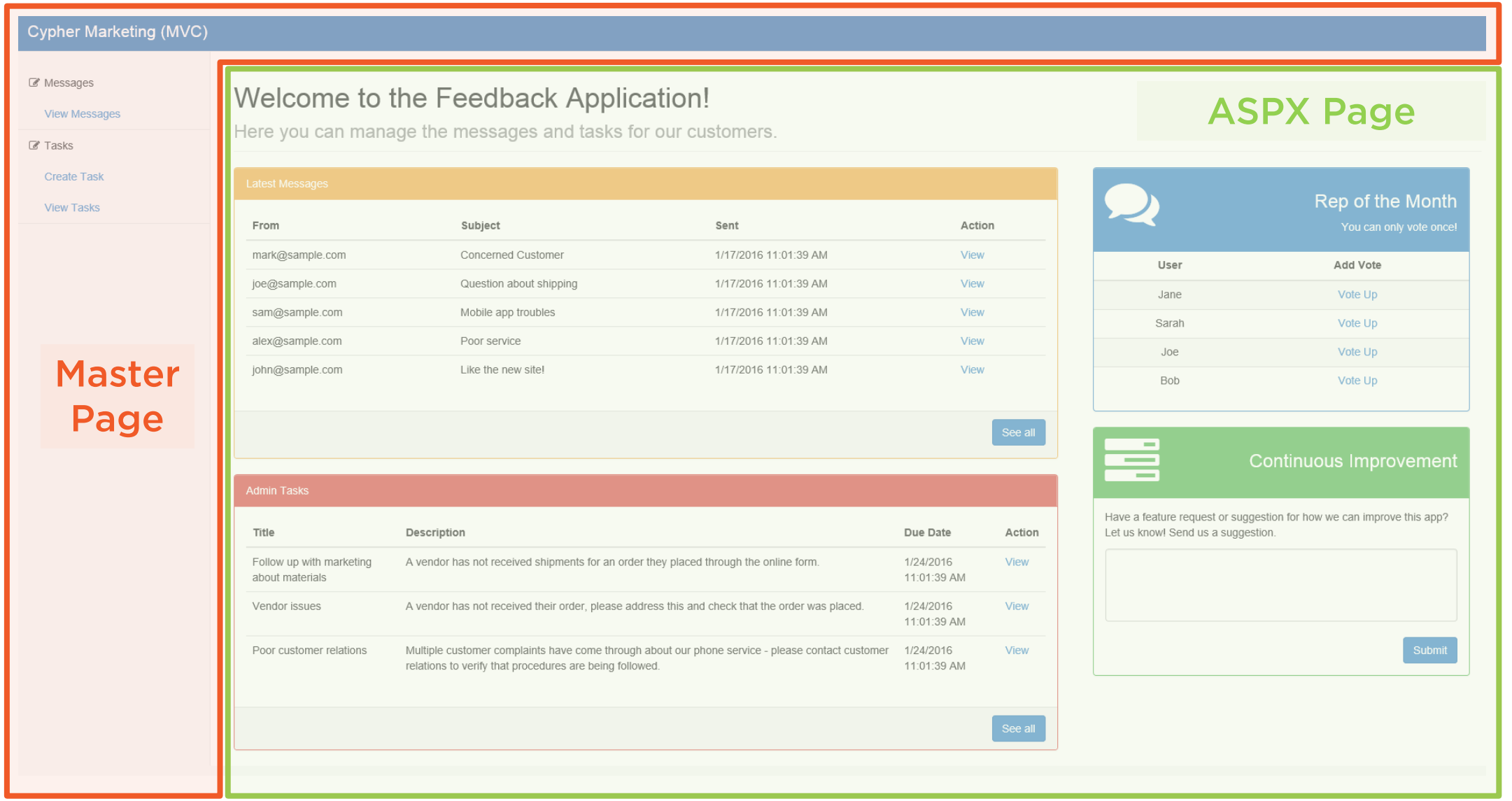
www.alexwolfthoughts.com

To-Do List

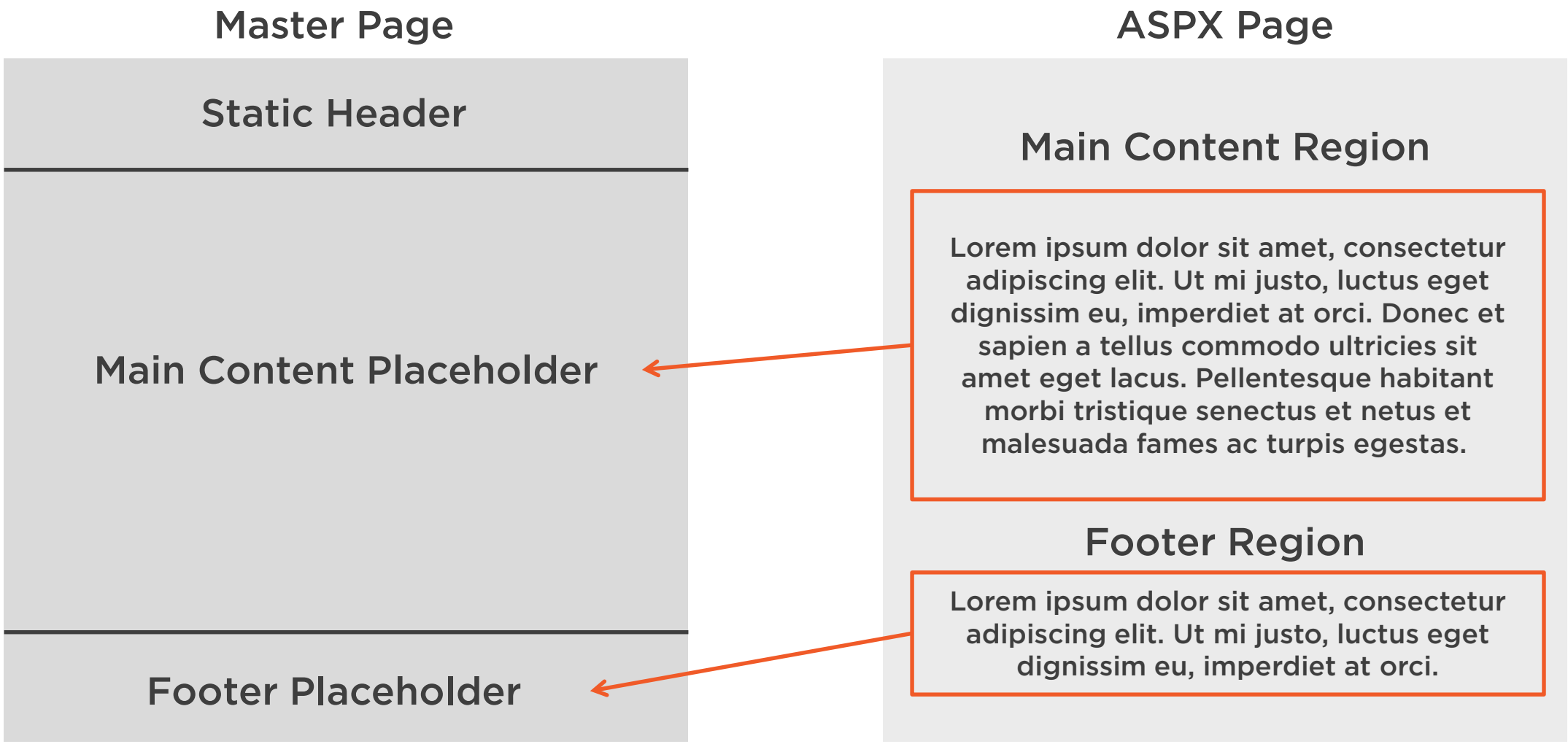
- Reviewing the Design Features of Web Forms
- Introducing Layouts and Views in MVC
- Demo: Building the Layout in MVC
- Demo: Adding Styles and Scripts in MVC
- Reviewing the Web Forms Server Controls
- Introducing the Razor View Engine
- Demo: Building the Navigation with Razor

Design Features of Web Forms

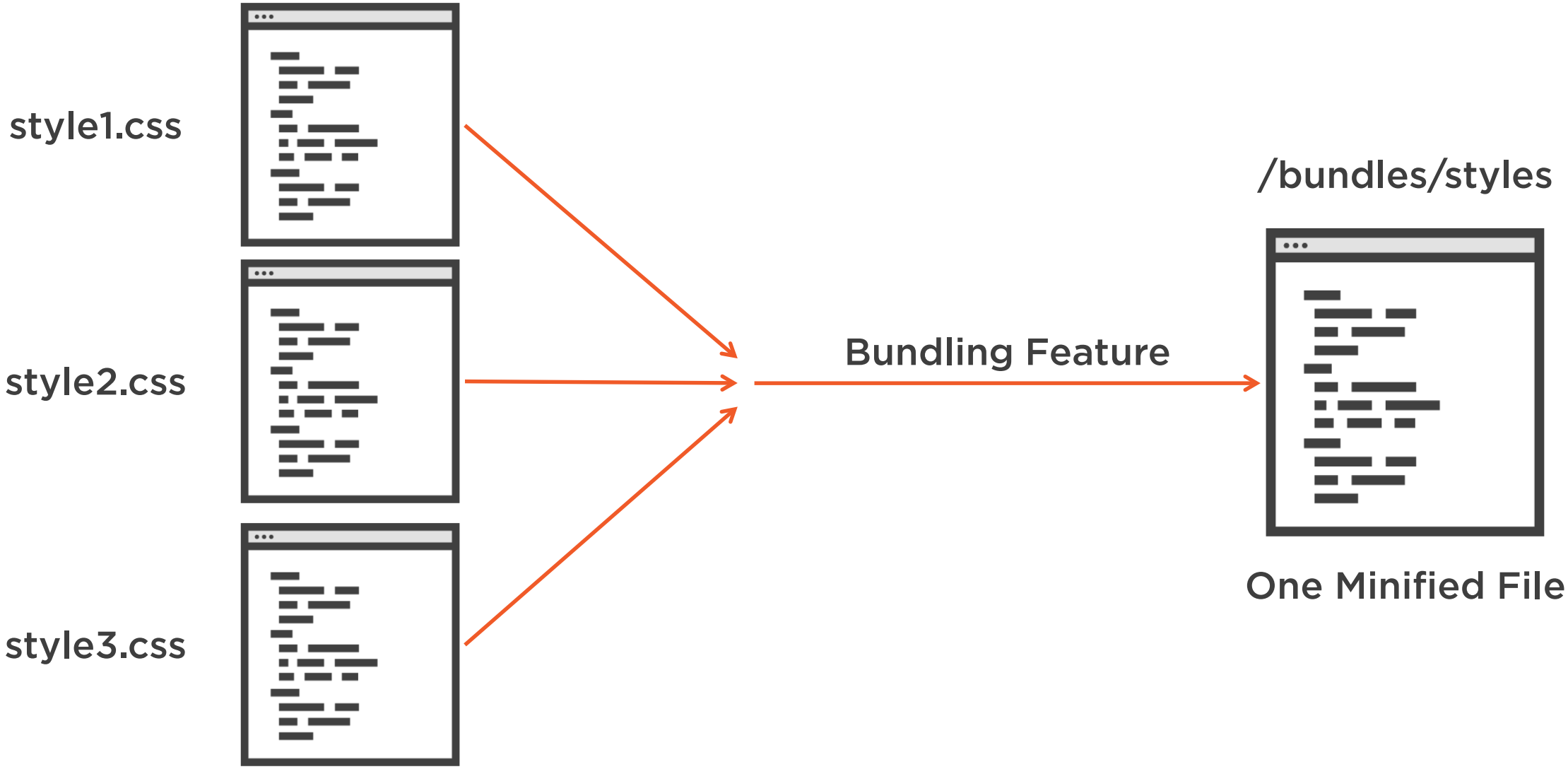
Understanding Master Pages



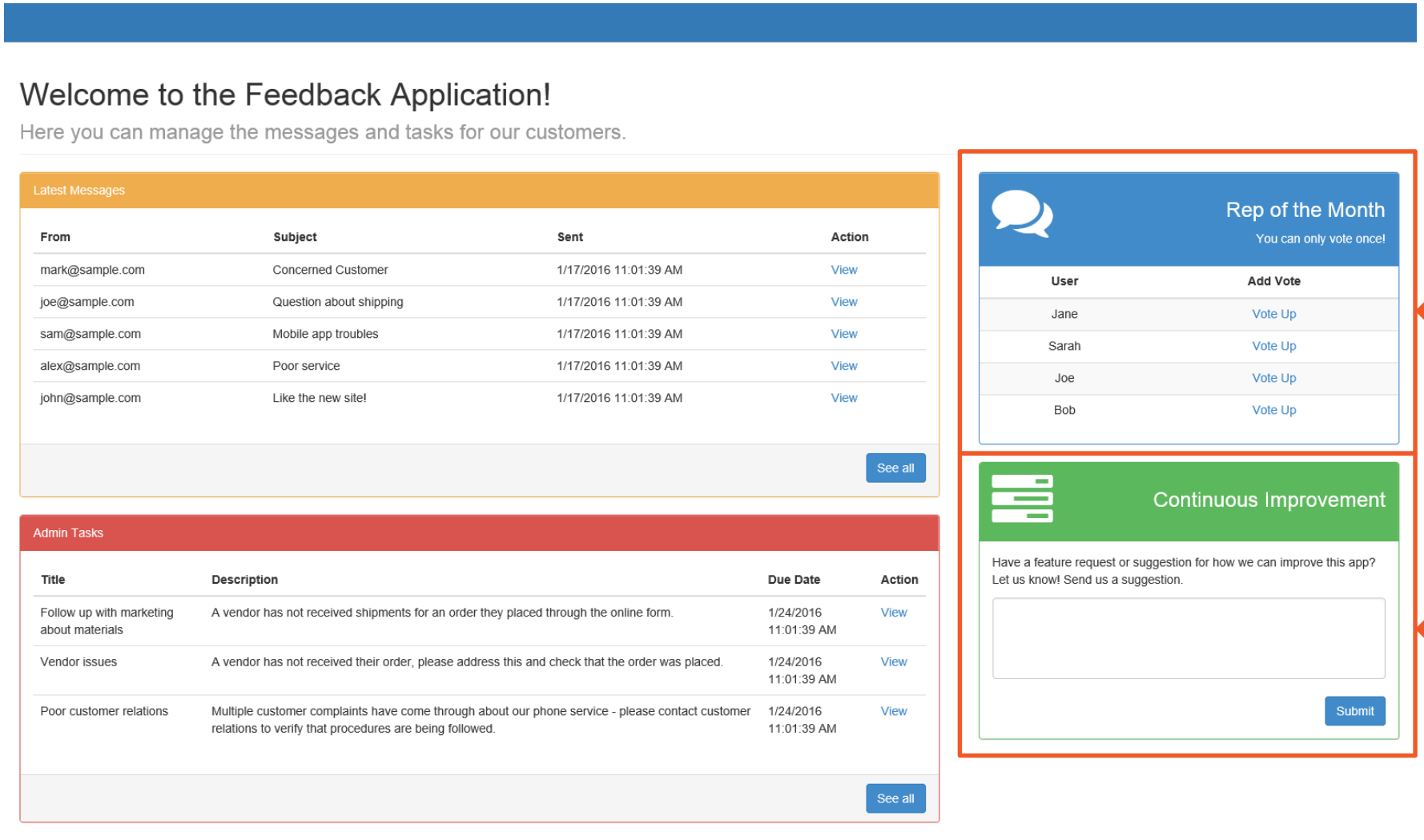
Master Page Content Regions



Scripts, Styles and Bundling



Organizing and Reusing with User Controls



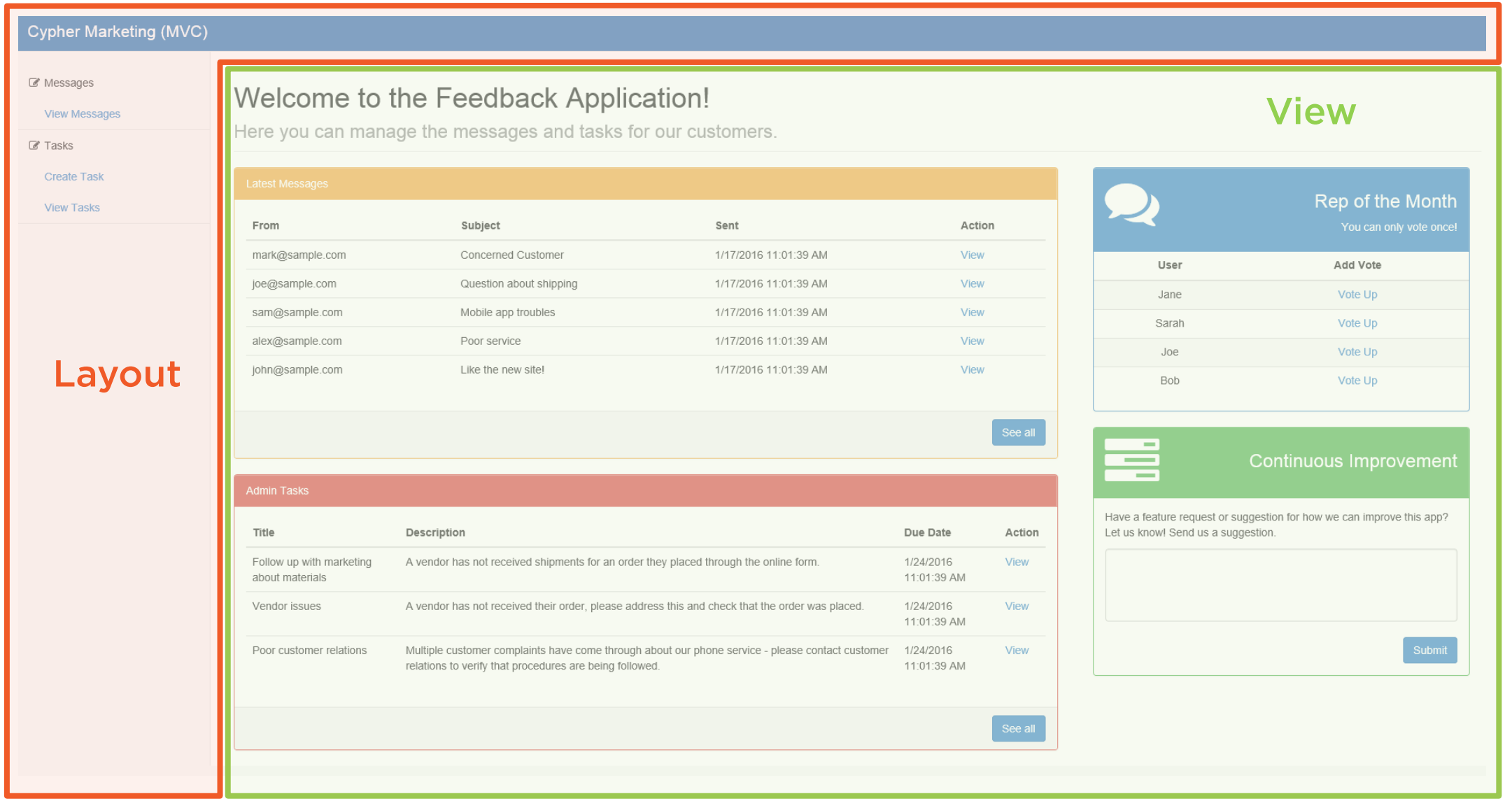
Survey Widget.ascx

Feedback Widget.ascx

Revisiting the Legacy Application Design

Layouts, Views, and Partial Views

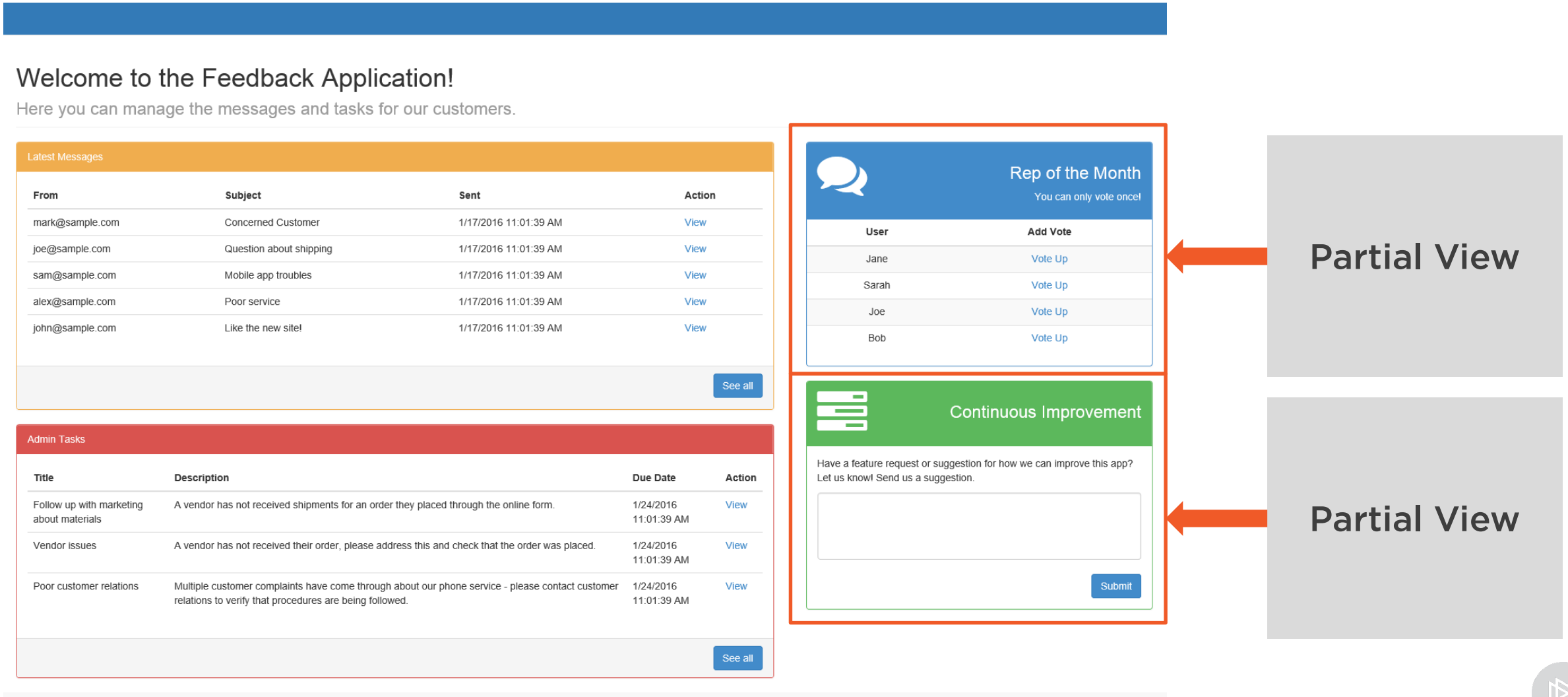
Understanding Layouts



```
<div id="wrapper">
        <div id="page-wrapper“>
                @RenderBody()
        </div>

@RenderSection("Secondary", false)

@if (!IsSectionDefined("Footer"))
{
<footer>This is the footer.</footer>
}
</div>
```

- Pulls the main view into the Layout
- Optional content section that can be overridden from view
- Optional footer content section that provides fallback content

Organizing and Reusing with User Controls



```
<div id="wrapper">

    <div id="widgets">

    @Html.Partial("SurveyWidget")

    @Html.Action("FeedbackWidget",
    "WidgetController")

    </div>

</div>
```

- Inserts a partial view directly
- Specifies an Action Method on a Controller to execute and inserts the result

Comparing Sections and Partial Views

Sections

Content placeholders in the Layout file that enforce document structure

Partial Views

Individual calls inside of Layouts or Views to inject the contents of an additional View

Comparing Views and Partial Views

Views

.cshtml file

Technically Equivalent

Partial Views

.cshtml file

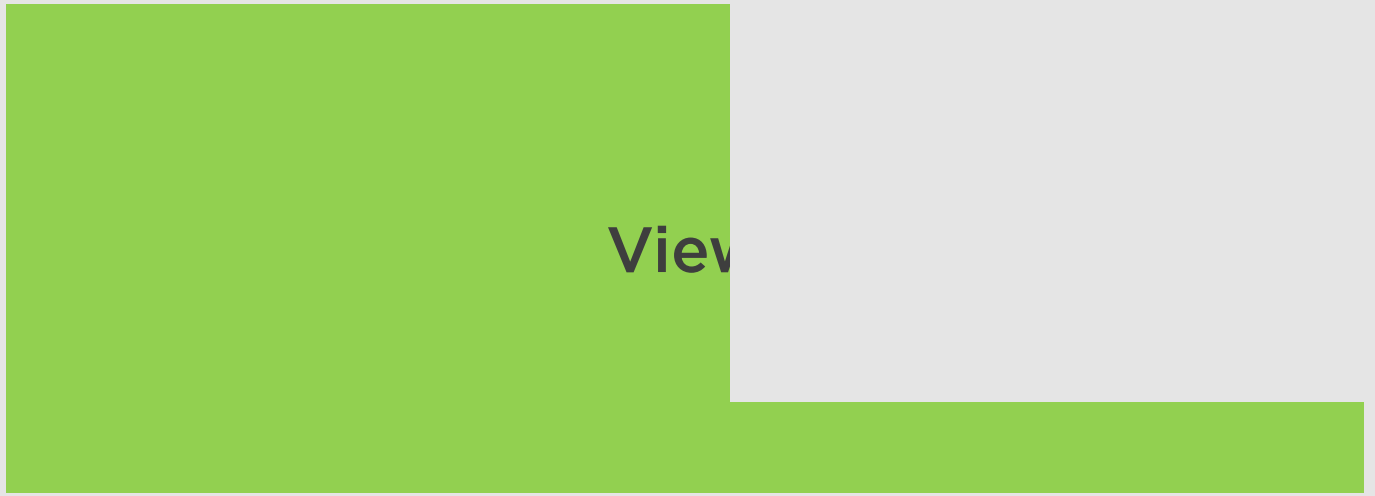
Generating Views and Partial Views

Regular Views	Partial Views
From a Controller	**From a Controller**
return View()	return PartialView()
From another View	**From another View**
Not Applicable	@Html.Action @Html.Partial

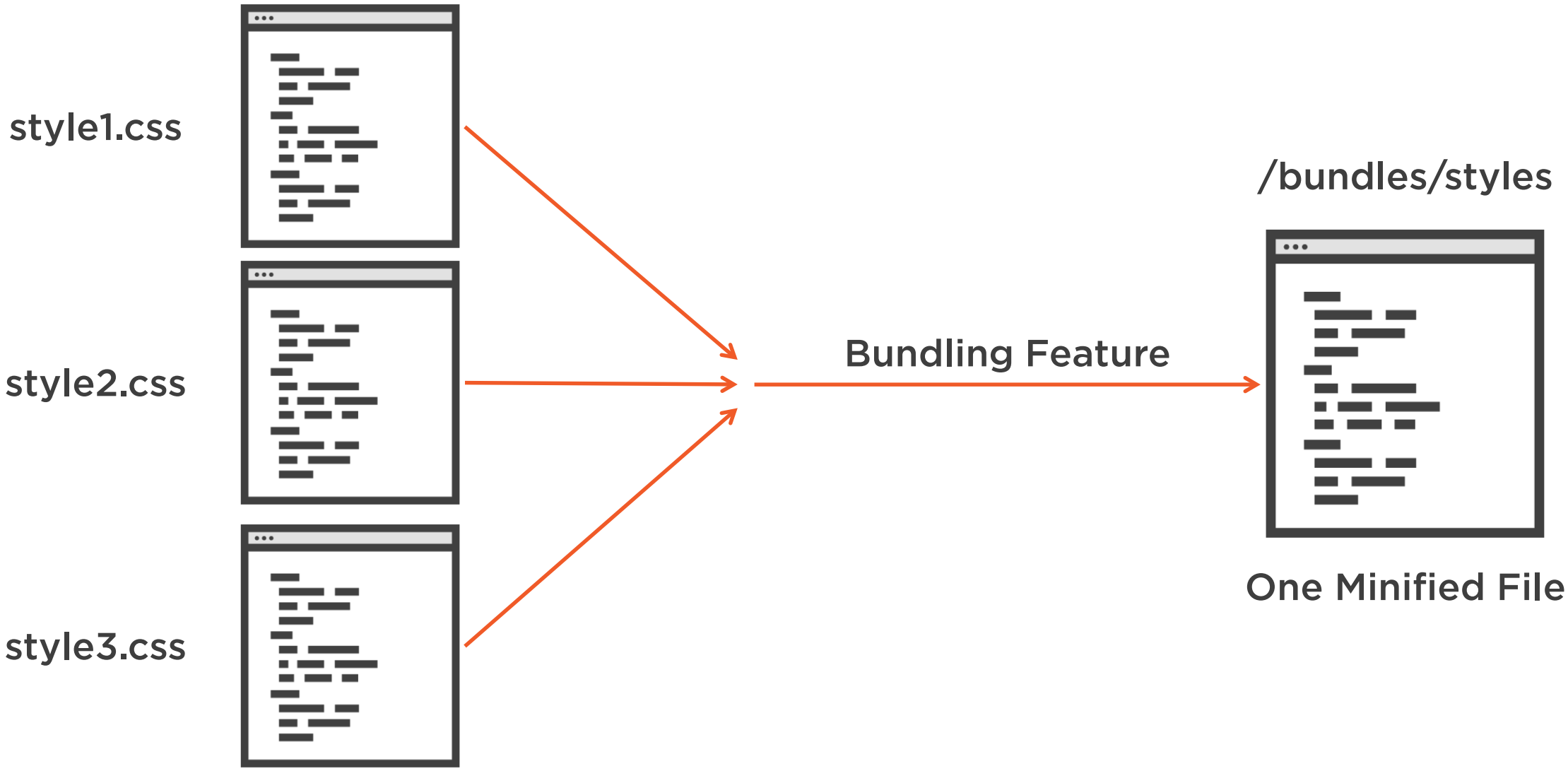
The ViewStart File

ViewStart

```
Layout = "~/Views/Shared/_La

//Other code and markup
```



Bundling in MVC



Building HTML Components in Web Forms

The Types of Server Controls

HTML Server Controls

Web Server Controls

Validation Controls

User Controls

```
<div class="wrapper">
    <label ID="subjectLabel" runat="server" for="subject"></label>
</div>
```

HTML Server Controls

Regular HTML empowered by Web Forms attributes

```
<div class=“wrapper”>

    <asp:TextBox ID="Subject" runat="server" CssClass="form-     control"></asp:TextBox>

    <asp:DropDownList CssClass="form-control" runat="server"
    ID="Category"></asp:DropDownList>

</div>
```

Web Server Controls

Abstraction over HTML, deeply integrated with Web Forms framework

```
<div class=“wrapper”>

    <asp:RequiredFieldValidator ID="DescriptionValidator"    ControlToValidate="Description“
runat="server">

    </asp:RequiredFieldValidator>

</div>
```

Validation Server Controls

Ensures valid data is contained in other Server Controls

User Controls

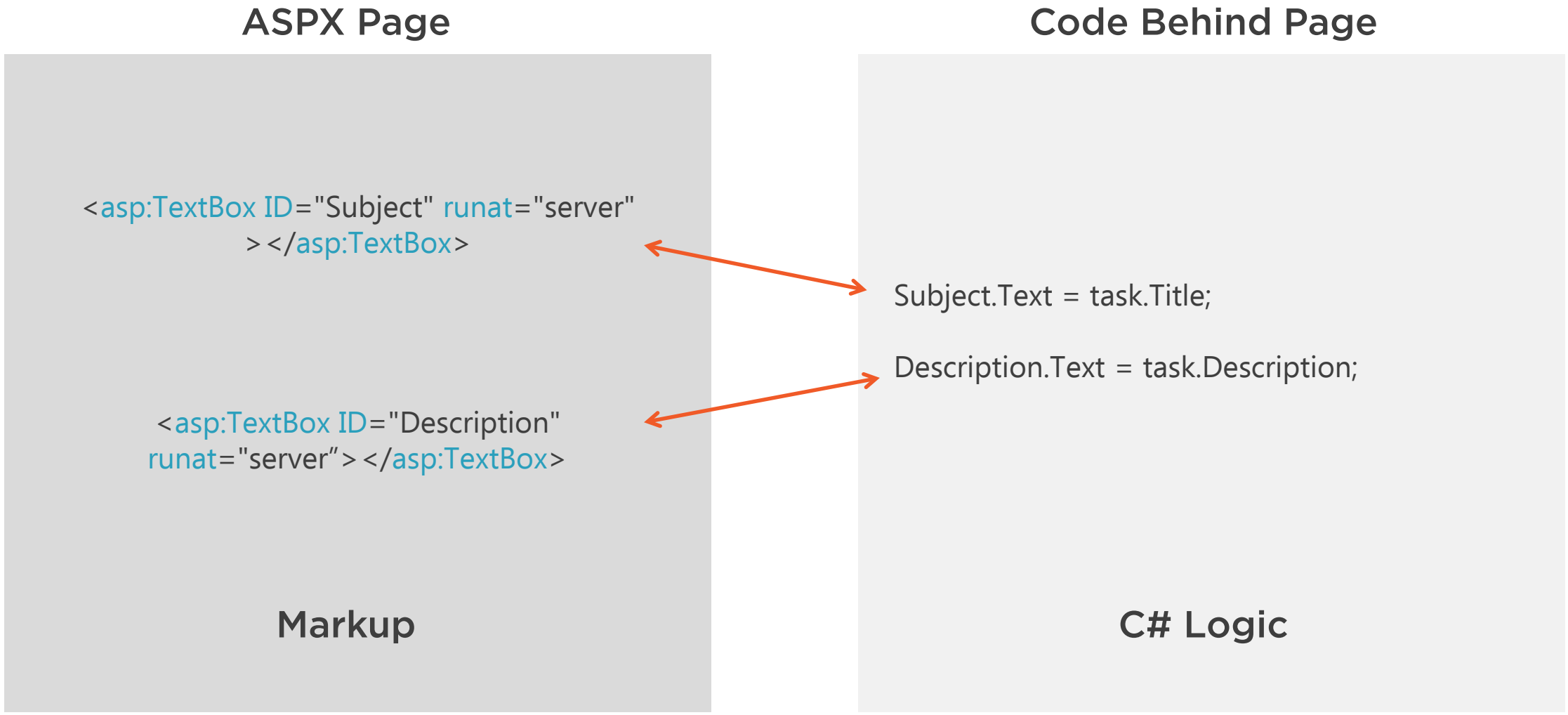
```
<div id="wrapper">

        <UserControls:Nav runat="server" id="Nav" />

</div>
```

- User Control embedded in an ASPX page

```
//Separate File

<div class="navbar-header">

        //Navigation markup

</div>
```

- User Control source file

Controlling HTML Through Code



The Razor View Engine

```
<div class="wrapper">
    @foreach(var message in messages){
        <span class="topic">@message.Subject</span>
    }
</div>
```

An Intelligent View Engine

Fluid transitions between HTML and Razor syntax

```
<div id="wrapper">


        <div id=“widgets“>

        @Html.ActionLink(“Create Task”,       “Create”,
“Task”)


        @Html.Partial(“SurveyWidget")


        @Url.Content("~/images/cat.png")

        </div>


</div>
```

- Renders a link to the create tasks page
- Inserts the contents inside of the SurveyWidget partial view
- Returns a string version of the URL to the specified item

Razor Helpers for Forms Elements

Razor Form Helper	HTML Equivalent
@Html.TextboxFor()	<Input type="text">
@Url.CheckboxFor()	<input type="checkbox">
@Url.DropDownListFor()	<select>[options]</select>

```
@model Task

<div id="wrapper">
    <div id="widgets">
    @Html.TextboxFor(m => m.Subject)

    @Html.TextboxFor(m => m.Date)

    @Html.CheckboxFor(m => m.IsDone)
    </div>
</div>
```

- The Model for the View and Form

- Strongly typed HTML helpers that render form fields for the Model properties

HTML Helpers generate
MVC friendly form fields.

```
<div id="wrapper">
        @foreach(var item in messages){
                <tr>
                        <td>@item.Subject</td>
                </tr>
        }
        @using(Html.BeginForm()) {
                //Form Contents
        }
        @if(@Model.IsDone){
                //Conditionally show
        }
</div>
```

- For Each loop to build elements
- Using statement for easy form creation
- Conditionally render markup

```
<div class="wrapper">
    @{
        var task = new Task();
        task.Subject = "Hello World";
        //Other Logic
    }
</div>
```

C# Code Blocks in Razor

Use with caution – is this really necessary?

Working with Razor

Summary

- Web Forms and MVC share similar design concepts, but different implementations
- Web Forms offers Master Pages, Pages and User Controls
- MVC provides Layouts, Views and Partial Views
- Web Forms uses Server Controls to work with HTML and framework features
- MVC offers lightweight but useful HTML Helpers through the Razor View Engine