

MVC 入门

《ASP.NET 程序设计》

主要内容

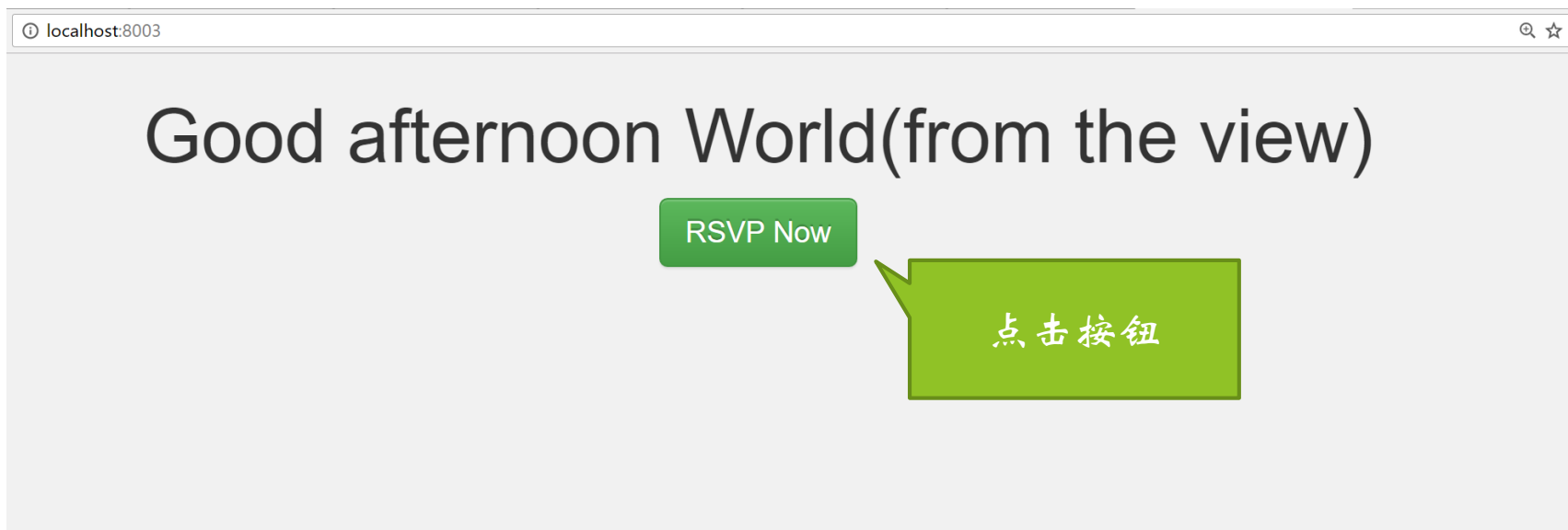
- ▶ 入门案例功能介绍
- ▶ 案例过程解析
- ▶ 开发过程回顾
- ▶ 本讲总结



入门案例功能介绍

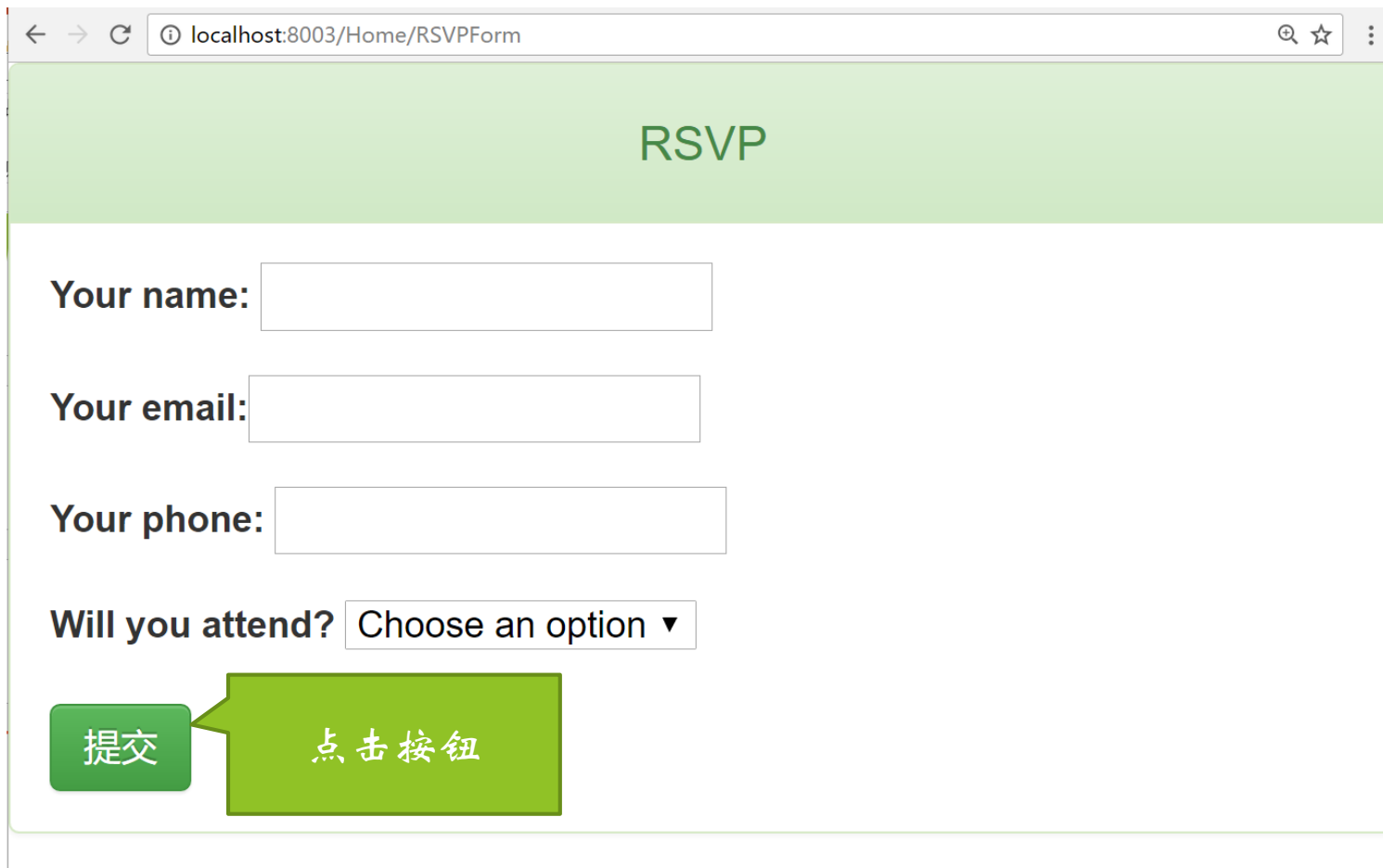
2 MVC入门

入门案例功能



项目启动页面

入门案例功能



localhost:8003/Home/RSVPForm

RSVP

Your name:

Your email:

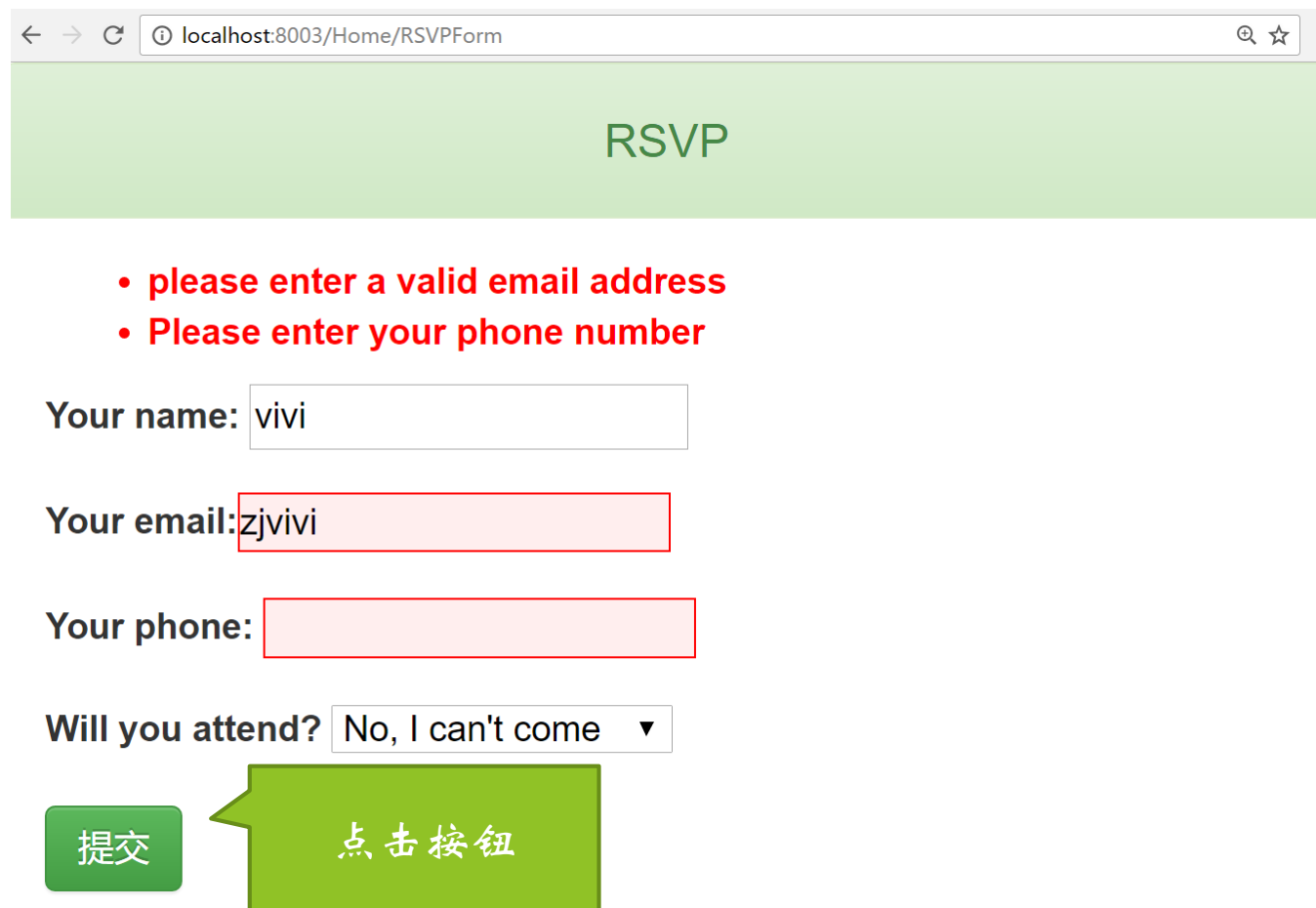
Your phone:

Will you attend?

点击按钮

RSVP邀请页面

入门案例功能



A screenshot of a web browser window displaying an RSVP form. The browser's address bar shows the URL "localhost:8003/Home/RSVPForm". The form has a light green header with the title "RSVP". Below the header, there are two red bullet points indicating validation errors: "• please enter a valid email address" and "• Please enter your phone number". The form contains four input fields: "Your name:" with the value "vivi", "Your email:" with the value "zjvivi", "Your phone:" which is empty, and "Will you attend?" with a dropdown menu showing "No, I can't come". A green "提交" (Submit) button is located at the bottom left of the form. A green speech bubble with the text "点击按钮" (Click button) points to the submit button.

localhost:8003/Home/RSVPForm

RSVP

- please enter a valid email address
- Please enter your phone number

Your name: vivi

Your email: zjvivi

Your phone:

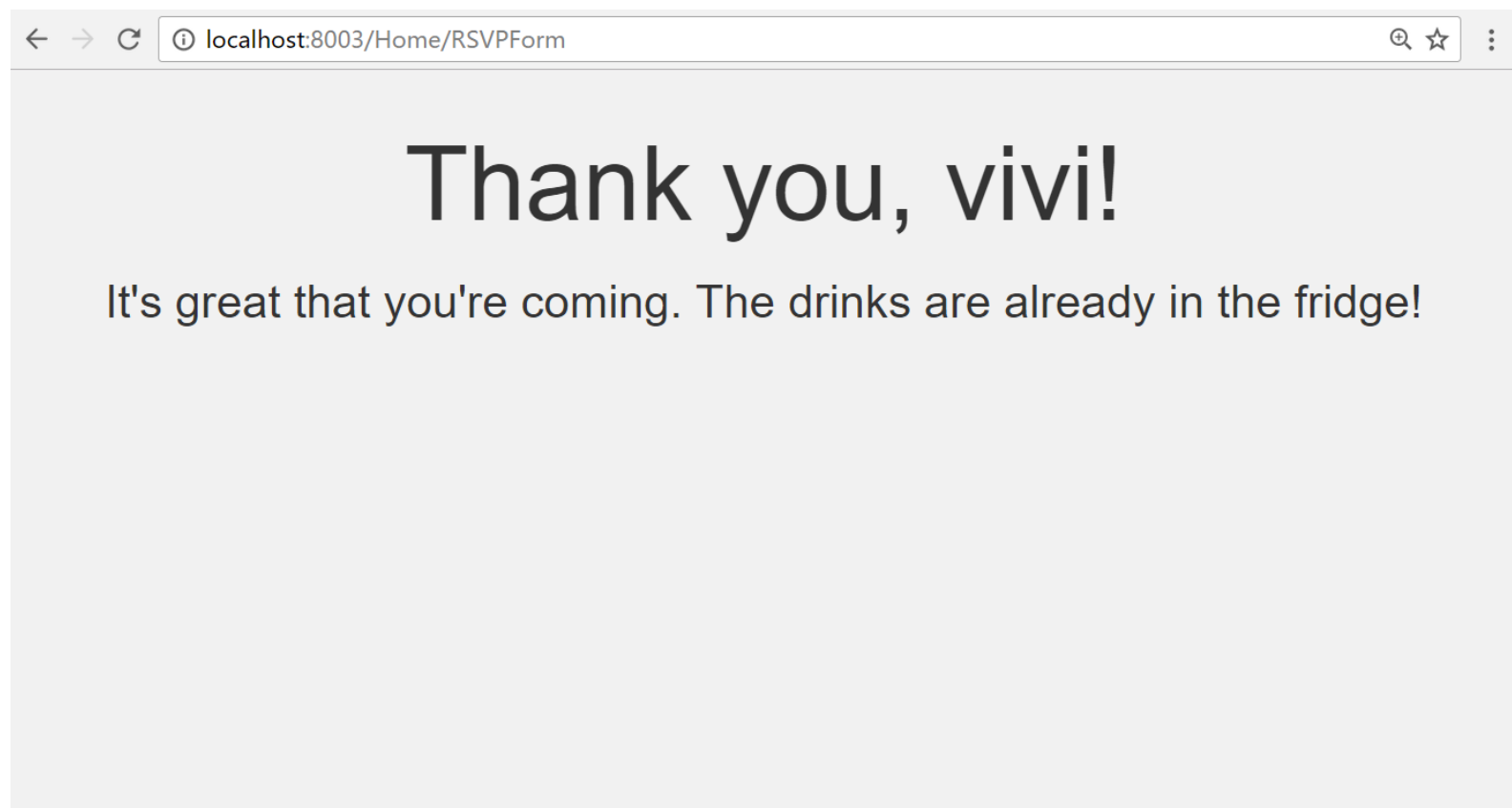
Will you attend? No, I can't come ▼

提交

点击按钮

错误提交后的表单校验提示

入门案例功能



正确提交后的结果页面

入门案例功能



正确提交后发送的邮件

案例过程解析

2 MVC入门

开发主要步骤介绍

1. 新建、保存打开工程
2. 新建和修改模型 (Model)
3. 新建和修改控制器 (Controller)
4. 新建和修改视图 (View)
5. 表单项的模型绑定
6. 表单项验证
7. Bootstrap引入使用
8. 表单错误提示样式处理
9. 邮件发信通知

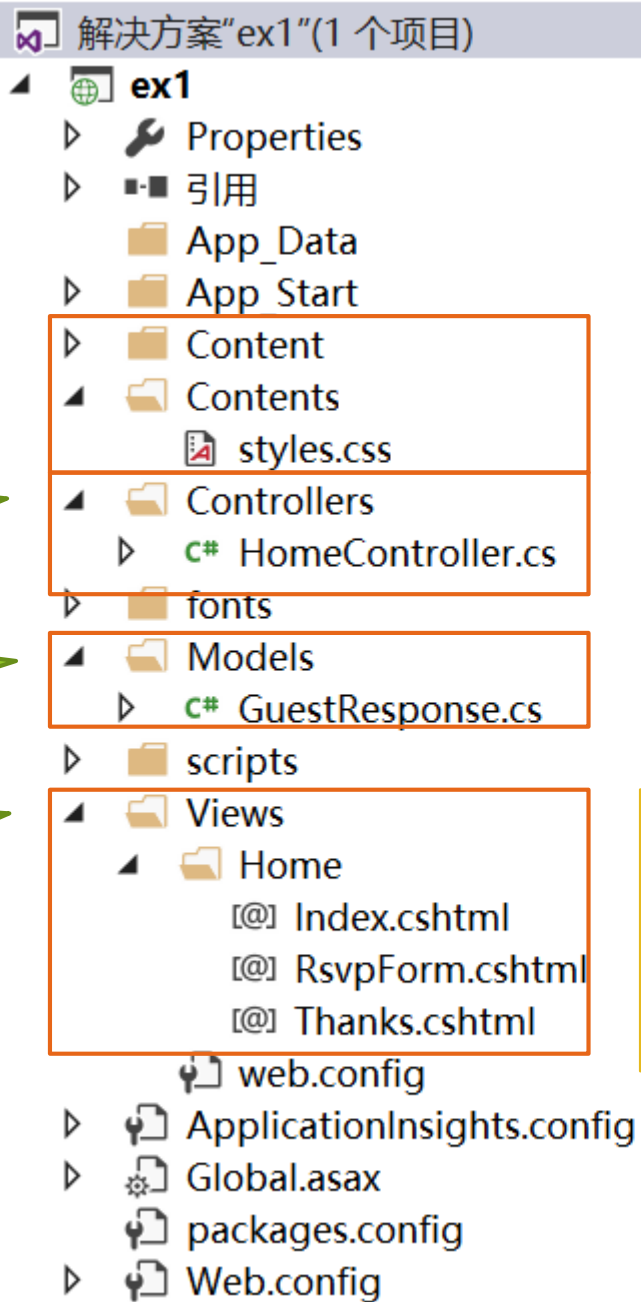
工程文件

样式文件 (Bootstrap框架)

控制器 Controllers

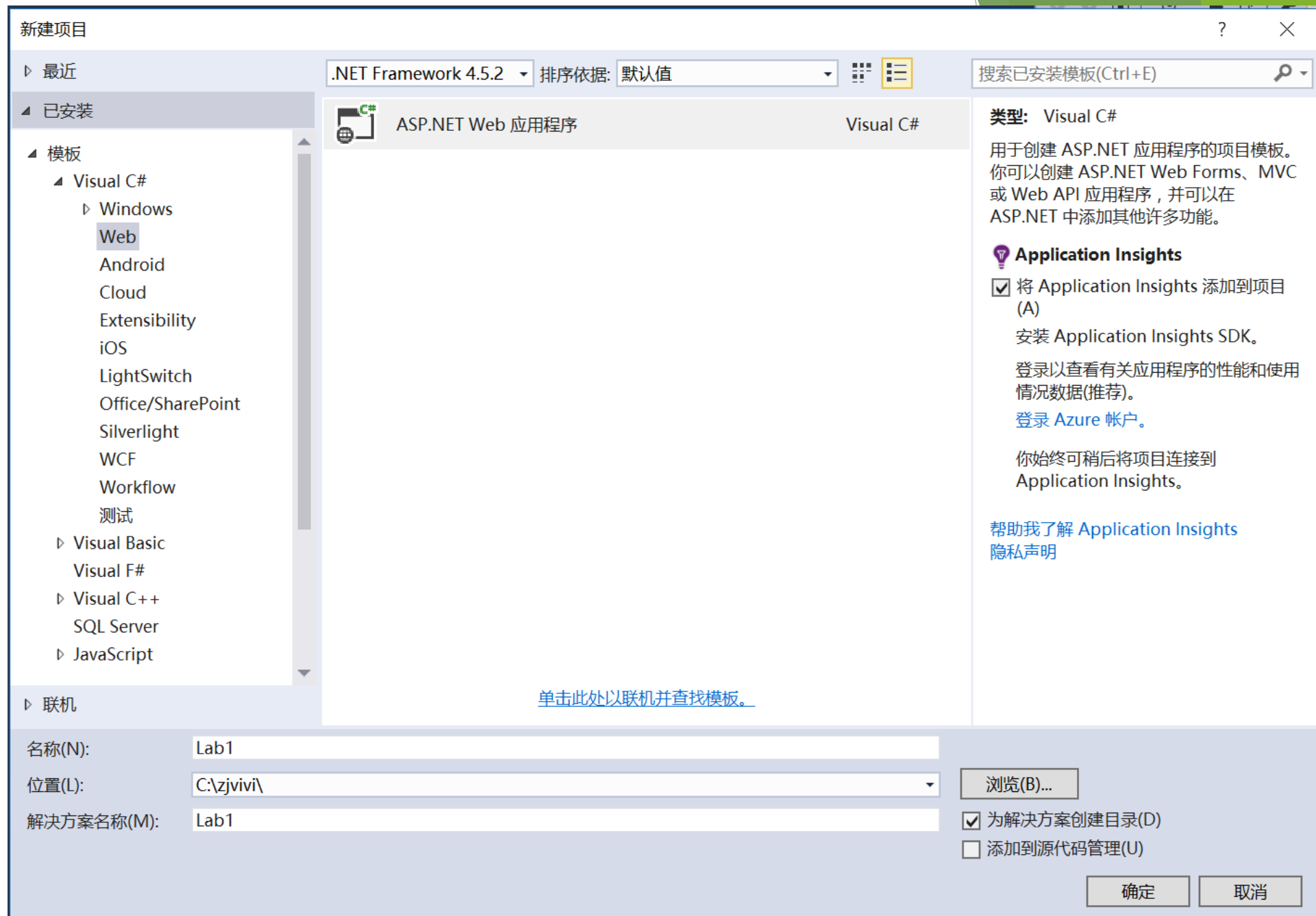
模型 Models

视图 Views



- Index.cshtml: 启动页
- RsvpForm.cshtml: 表单页
- Thanks.cshtml: 反馈页

新建工程



新建工程

新建 ASP.NET 项目 - Lab1

选择模板(S):

ASP.NET 4.5.2 模板

Empty

Web Forms

MVC

Web API

Single Page Application

Azure API App (Preview)

Azure Mobile Service

ASP.NET 5 模板

Get ASP.NET 5 RC

为以下项添加文件夹和核心引用:

☐ Web Forms

☒ MVC

☐ Web API

☐ 添加单元测试(U)

测试项目名称(T): Lab1.Tests

用于创建 ASP.NET 应用程序的空项目模板。此模板中没有任何内容。

[了解更多](#)

更改身份验证(A)

身份验证: 不进行身份验证

Microsoft Azure

Host in the cloud

Web App

确定

取消

新建 Controller

1. 在“解决方案管理浏览器”窗口，右击“Controllers”。
2. 选择“添加”-->“控制器...”。
3. 选择“MVC5控制器-空”。
4. 单击确定后，输入控制器名称“HomeController.cs”。



Controller的取名要和“App_Start/RouteConfig.cs”中的设定相一致。

修改 IndexController

新建后的controller

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.Mvc;
6
7 namespace Lab1.Controllers
8 {
9     0 个引用
10    public class IndexController : Controller
11    {
12        // GET: Index
13        0 个引用
14        public ActionResult Index()
15        {
16            return View();
17        }
18    }
19 }
```

运行后，404错误

修改后的controller

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.Mvc;
6
7 namespace Lab1.Controllers
8 {
9     0 个引用
10    public class HomeController : Controller
11    {
12        // GET: Index
13        0 个引用
14        public string Index()
15        {
16            return "Hello World!";
17        }
18    }
19 }
```

新建 View

1. 在“解决方案管理浏览器”窗口，右击“Views”。
2. 选择“添加”-->“视图...”。
3. 输入视图名称“index”，模板为“Empty”，不使用布局页。
4. 同理，可以新建其他视图页。



添加视图

视图名称(N): View

模板(T): Empty (不具有模型)

模型类(M):

选项:

☐ 创建为分部视图 (C)

☐ 引用脚本库(R)

☐ 使用布局页(U):

(如果在 Razor_viewstart 文件中设置了此选项，则留空)

添加 取消

启动页的取名要和“App_Start/RouteConfig.cs”中的设定相一致。

建立 Controller 和 View 的关系

```
public ActionResult Index()  
{  
    int hour = DateTime.Now.Hour;  
    ViewBag.Greeting = hour < 12 ? "Good Morning" : "Good afternoon";  
    return View();  
}
```

1. 修改转向类型为：ActionResult，并且修改Index方法中的代码如下。
2. ViewBag为视图包对象，是Controller基类中的一个成员，将数据从控制器传递给视图。它是一种动态对象，可任意赋值属性，属性的值在视图中可用。

建立 Controller 和 View 的关系

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

```
@{
    Layout = null;
}

<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Index</title>
    <style>
        .btn a{color:white; text-decoration:none;}
        body{background-color:#f1f1f1;}
    </style>
</head>
<body>
    <div class="text-center">
        <h1>
            @ViewBag.Greeting World(from the view)
        </h1>
    </div>
</body>
</html>
```

MVC5中视图采用Razor引擎。
Razor引擎处理视图内容并生成
发送给浏览器的HTML。

此处表示不用页面不用布局模板。

调用Controller中定义的值

建立 Controller 和 View 的关系

```
routes.MapRoute(  
    name: "Default",  
    url: "{controller}/{action}/{id}",  
    defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }  
);
```

“App_Start/RouteConfig.cs”
中定义的启动路由

路由决定了controllers和views
中启动项的命名规则

建立 Controller 和 View 的关系

```
public class HomeController : Controller
```

```
{
```

```
// GET: Home
```

0 个引用

```
public ActionResult Index()
```

```
{
```

```
    int hour = DateTime.Now.Hour;
```

```
    ViewBag.Greeting = hour < 12 ? "Good Morning" : "Good afternoon";
```

```
    return View();
```

```
}
```

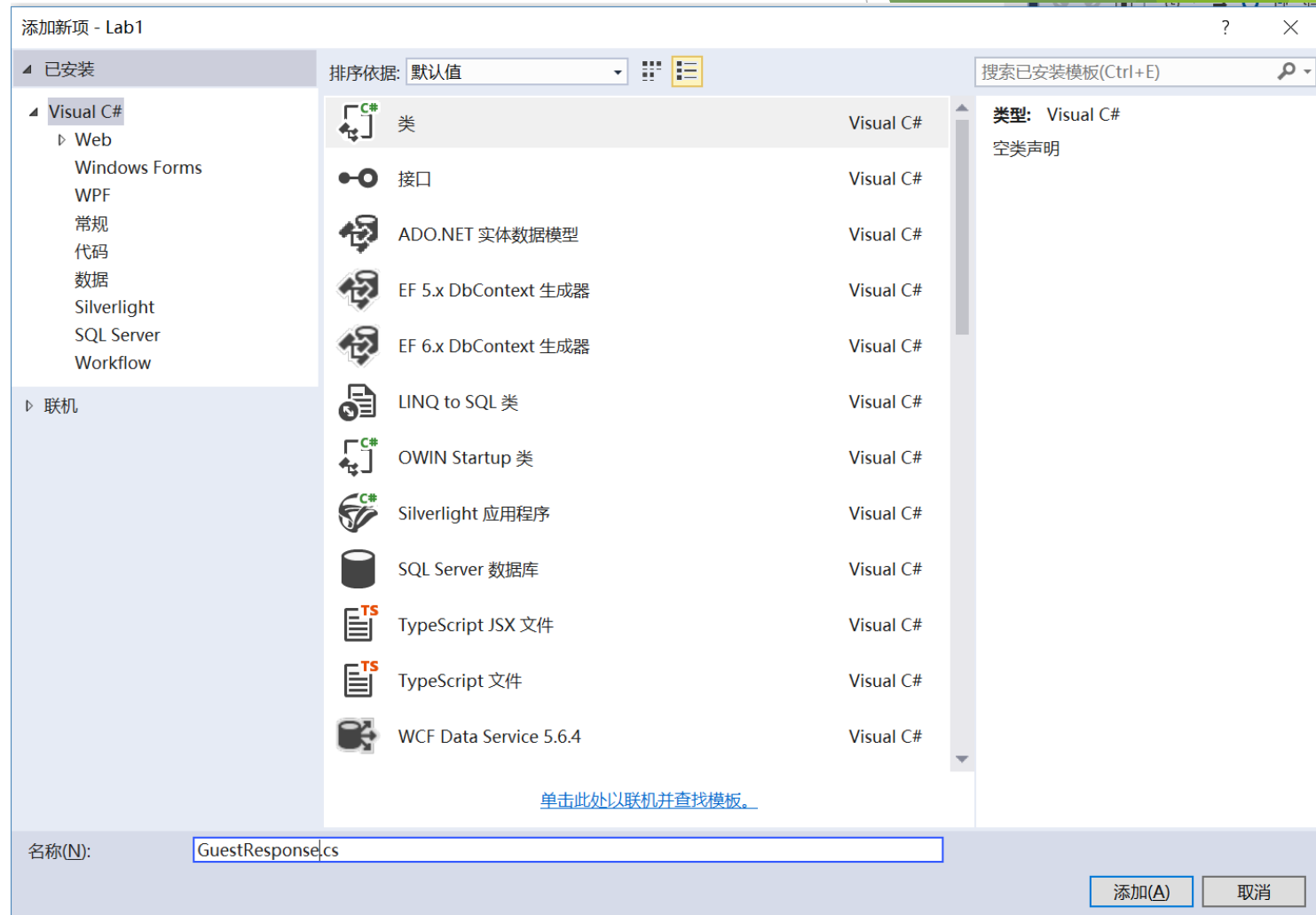
Home控制器中的action是
“index”的方法为启动动作。

```
└─ Views
   └─ Home
      └─ [@] Index.cshtml
```

Views中的文件目录，和
controller相对应。

新建 Model

1. 在“解决方案管理浏览器”窗口，右击“Models”。
2. 选择“添加”-->“类...”。
3. 输入类名称
“GuestResponse.cs”。



Model

(GuestResponse.cs)

的 源 代 码

3 个引用

```
public class GuestResponse  
{
```

1 个引用

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

0 个引用

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

0 个引用

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

1 个引用

```
public bool? willAttend { get; set; }
```

```
}
```

引用数是系统自动更新的，
默认从0开始。

Index视图中添加“RSVP”的跳转按钮

```
<div >
  <h1>
    @ViewBag.Greeting World(from the view)
  </h1>
  <div >
    @Html.ActionLink("RSVP Now", "RSVPForm")
  </div>
</div>
```

HTML.ActionLink是HTML的辅助器方法之一。

参数1:该链接的显示文本。

参数2:单击链接后执行的动作,即controller中的动作方法名。

RsvpForm视图的源代码

以下都是HTML辅助器用于渲染成HTML代码:

- HTML.BeginForm()
- HTML.TextBoxFor()
- HTML.DropDownListFor()

```
<body>
  <div>
    <div>
      <h4>RSVP</h4>
    </div>
    <div>
```

```
      @using (Html.BeginForm())
```

```
{
```

```
    @Html.ValidationSummary()
```

```
    <div><label>Your name:</label> @Html.TextBoxFor(x => x.Name)</div>
```

```
    <div><label>Your email:</label> @Html.TextBoxFor(x => x.Email)</div>
```

```
    <div>
```

```
        <label>Your phone:</label> @Html.TextBoxFor(x => x.Phone)
```

```
    </div>
```

```
    <div>
```

```
        <label>Will you attend?</label>
```

```
        @Html.DropDownListFor(x => x.willAttend, new[] {
```

```
            new SelectListItem() {Text="No, I can't come", Value= bool.FalseString },
```

```
            new SelectListItem() {Text="Yes, I'll be there",Value=bool.TrueString }
```

```
        }, "Choose an option")
```

```
    </div>
```

```
    <input type="submit" value="提交" />
```

```
}
```

```
</div>
```

```
</div>
```

```
</body>
```

```
@model Lab1.Models.GuestResponse
```

页面顶部添加model的引用

Lambda表达式

HomeController中添加“RsvpForm”的动作

[HttpGet]

0 个引用

```
public ActionResult RsvpForm()  
{  
    return View();  
}
```

根据用户的不同请求来设置不同动作。

这个动作负责显示最初的空白表单

[HttpPost]

0 个引用

```
public ActionResult RsvpForm(GuestResponse guestResponse)  
{  
    if (ModelState.IsValid)  
    {  
        return View("Thanks", guestResponse);  
    }  
    else {  
        return View();  
    }  
}
```

Controller中要引用模型，即“using ex1.Models;”

这个动作负责显示最初的空白表单接收所递交的数据，并决定用它做什么。

@model Lab1.Models.GuestResponse

Thanks视图的源代码

页面顶部添加model的引用

```
<div >
  <h1>Thank you, @Model.Name!</h1>
  <div>
    @if (Model.willAttend == true)
    {
      @:It's great that you're coming. The drinks are already in the fridge!
    }
    else
    {
      @:Sorry to hear that you can't make it, but thanks for letting us know.
    }
  </div>
</div>
```

使用模型绑定
@Model.Name等为引用
Model中的属性值

添加单证表验证

```
public class GuestResponse
{
    [Required(ErrorMessage = "Please enter your name")]
    2 个引用
    public string Name { get; set; }
    [Required(ErrorMessage = "Please enter your email")]
    [RegularExpression(".+\\@.+\\..+",
        ErrorMessage = "please enter a valid email address")]
    1 个引用
    public string Email { get; set; }
    [Required(ErrorMessage = "Please enter your phone number")]
    1 个引用
    public string Phone { get; set; }
    [Required(ErrorMessage = "Please specify whether you'll attend")]
    2 个引用
    public bool? willAttend { get; set; }
}
```

如果ModelState.IsValid为真表示验证通过。几个引用是在实际发生后自动添加的，无需修改。

错误摘要显示

```
<div class="panel-body">
  @using (Html.BeginForm())
  {
    @Html.ValidationSummary()
    <div class="form-group"><label>Your name:</label> @Html.TextBoxFor(x => x.Name)</div>
    <div class="form-group"><label>Your email: </label> @Html.TextBoxFor(x => x.Email)</div>
    <div class="form-group">
      <label>Your phone:</label> @Html.TextBoxFor(x => x.Phone)
    </div>
    <div class="form-group">
      <label>Will you attend?</label>

      @Html.DropDownListFor(x => x.willAttend, new[] {
        new SelectListItem() {Text="No, I can't come", Value= bool.FalseString },
        new SelectListItem() {Text="Yes, I'll be there",Value=bool.TrueString }
      }, "Choose an option")
    </div>
    <input type="submit" value="提交" class="btn btn-success" />
  }
</div>
```

错误项高亮显示

```
.field-validation-error{color:red;}  
.field-validation-valid{display:none;}  
.input-validation-error{border:1px solid #f00;background-color:#fee;}  
.validation-summary-errors{font-weight:bold;color:#f00;}  
.validation-summary-valid{display:none;}
```

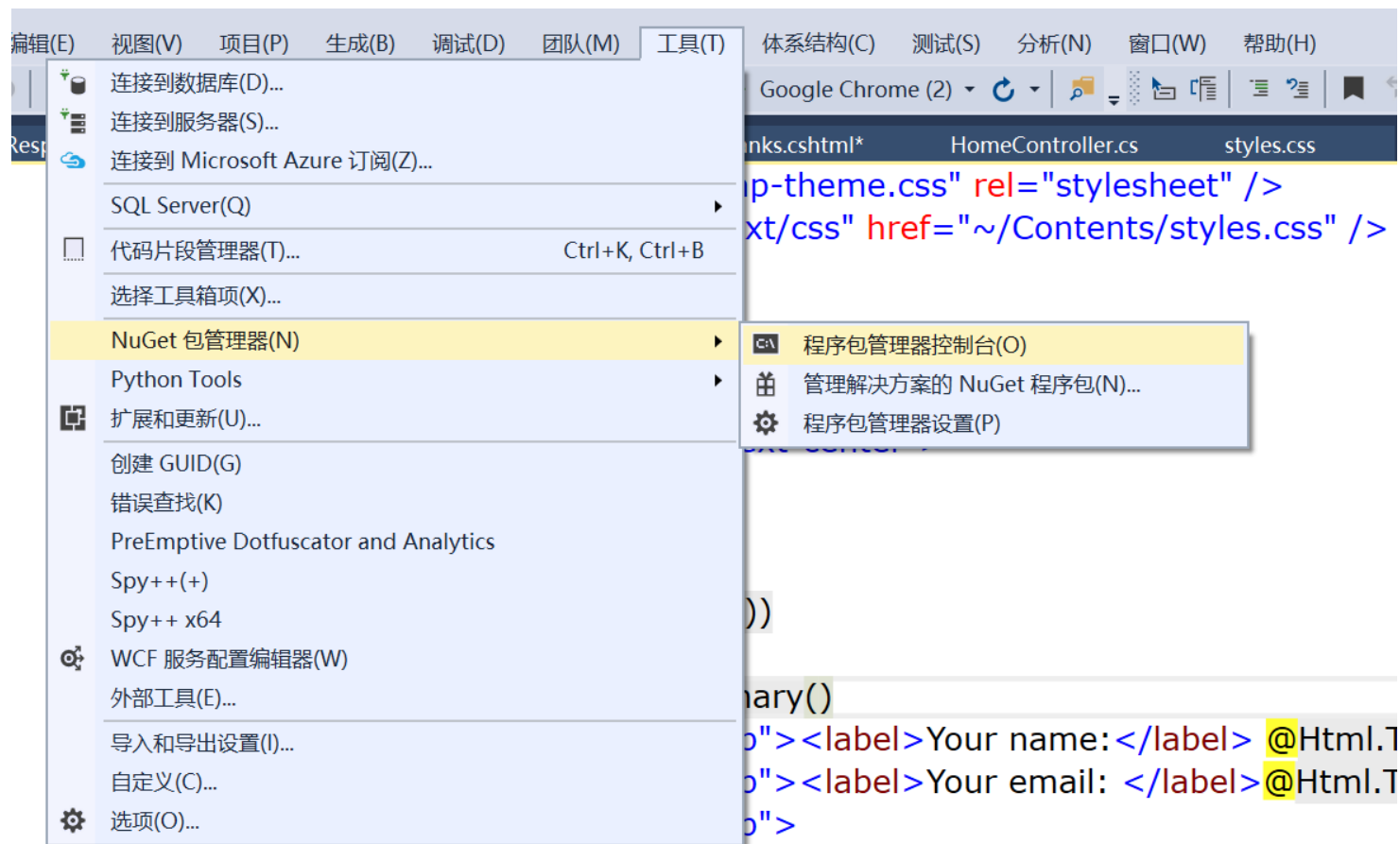
新建“Contents/Style.css”文件，
可在运行页面的源代码中查看到对应项
的class名，再自行设置。

```
<link rel="stylesheet" type="text/css" href="~/Contents/styles.css" />
```

将CSS文件引入到RsvpForm.cshtml中

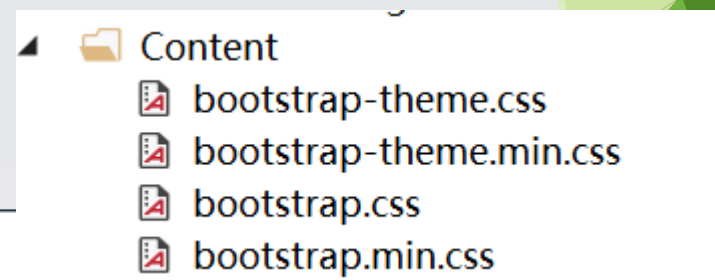
美化页面

1、利用NuGet工具引入Bootstrap框架



美化页面

1、利用NuGet工具引入Bootstrap框架



美化页面

2、修改Index.cshtml页面

```
<head>
  <meta name="viewport" content="width=device-width" />
  <title>Index</title>
  <link href="~/Content/bootstrap.css" rel="stylesheet"/>
  <link href="~/Content/bootstrap-theme.css" rel="stylesheet" />
  <style>
    .btn a{color:white; text-decoration:none;}
    body{background-color:#f1f1f1;}
  </style>
</head>
<body>
  <div class="text-center">
    <h1>
      @ViewBag.Greeting World(from the view)
    </h1>
    <div class="btn btn-success">
      @Html.ActionLink("RSVP Now", "RSVPForm")
    </div>
  </div>
</body>
```


美化页面

2、修改 RsvpForm. cshtml页面

```
<body>
  <div class="panel panel-success">
    <div class="panel-heading text-center">
      <h4>RSVP</h4>
    </div>
    <div class="panel-body">
      @using (Html.BeginForm())
      {
        @Html.ValidationSummary()
        <div class="form-group"><label>Your name:</label> @Html.TextBoxFor(x => x.Name)</div>
        <div class="form-group"><label>Your email: </label> @Html.TextBoxFor(x => x.Email)</div>
        <div class="form-group">
          <label>Your phone:</label> @Html.TextBoxFor(x => x.Phone)
        </div>
        <div class="form-group">
          <label>Will you attend?</label>

          @Html.DropDownListFor(x => x.willAttend, new[] {
            new SelectListItem() {Text="No, I can't come", Value= bool.FalseString },
            new SelectListItem() {Text="Yes, I'll be there",Value=bool.TrueString }
          }, "Choose an option")
        </div>
        <input type="submit" value="提交" class="btn btn-success" />
      }
    </div>
  </div>
</body>
```

美化页面

2、修改

Thanks.cshtml 页面

```
<div class="text-center">
  <h1>Thank you, @Model.Name!</h1>
  <div class="lead">
    @if (Model.willAttend == true)
    {
      @:It's great that you're coming. The drinks are already in the fridge!
    }
    else
    {
      @:Sorry to hear that you can't make it, but thanks for letting us know.
    }
  </div>
</div>
```

```
<link href="~/Content/bootstrap.css" rel="stylesheet" />
<link href="~/Content/bootstrap-theme.css" rel="stylesheet" />
<title>Thanks</title>
<style>
  body{background:#f1f1f1;}
</style>
```

发送反馈邮件

WebMail的具体用法请见:

[https://msdn.microsoft.com/en-us/library/system.web.helpers.webmail\(v=vs.111\).aspx](https://msdn.microsoft.com/en-us/library/system.web.helpers.webmail(v=vs.111).aspx)

```
@{  
    try  
    {  
        WebMail.SmtpServer = "smtp.126.com";  
        WebMail.SmtpPort = 25;  
        WebMail.EnableSsl = true;  
        WebMail.UserName = "web_dev_edu_hk@126.com";  
        WebMail.Password = "vivi123";  
        WebMail.From = "web_dev_edu_hk@126.com";  
        WebMail.Send("zjvivi@126.com", "RSVP Notification", "Notification coming!");  
    }  
    catch(Exception)  
    {  
        @: <b>Sorry- we couldn't send the email to confirm yo  
    }  
}
```

设置发送邮件服务器

设置发送人信息

Send方法参数介绍:

参数1:接收人邮箱

参数2: 发送主题

参数3: 发送内容

开发过程回顾

2 MVC入门

你言我语



本讲总结

2 MVC入门

主要知识点

- ▶ 理解MVC架构以及它们之间的分工合作。
- ▶ Razor视图引擎
- ▶ 路由设置
- ▶ HTML辅助器使用
- ▶ Lamda表达式
- ▶ 表单验证
- ▶ 使用模型绑定
- ▶ Bootstrap引入和应用
- ▶ Html页面渲染、传输和呈现的过程了解
- ▶ WebMail对象的使用

作业

- ▶ 安装VS 2013 Express for Web及以上（本人安装VS 2015）
- ▶ 完成“实验1 MVC入门案例”。
- ▶ 准备下周讲解自己的实验1任务。

The end!