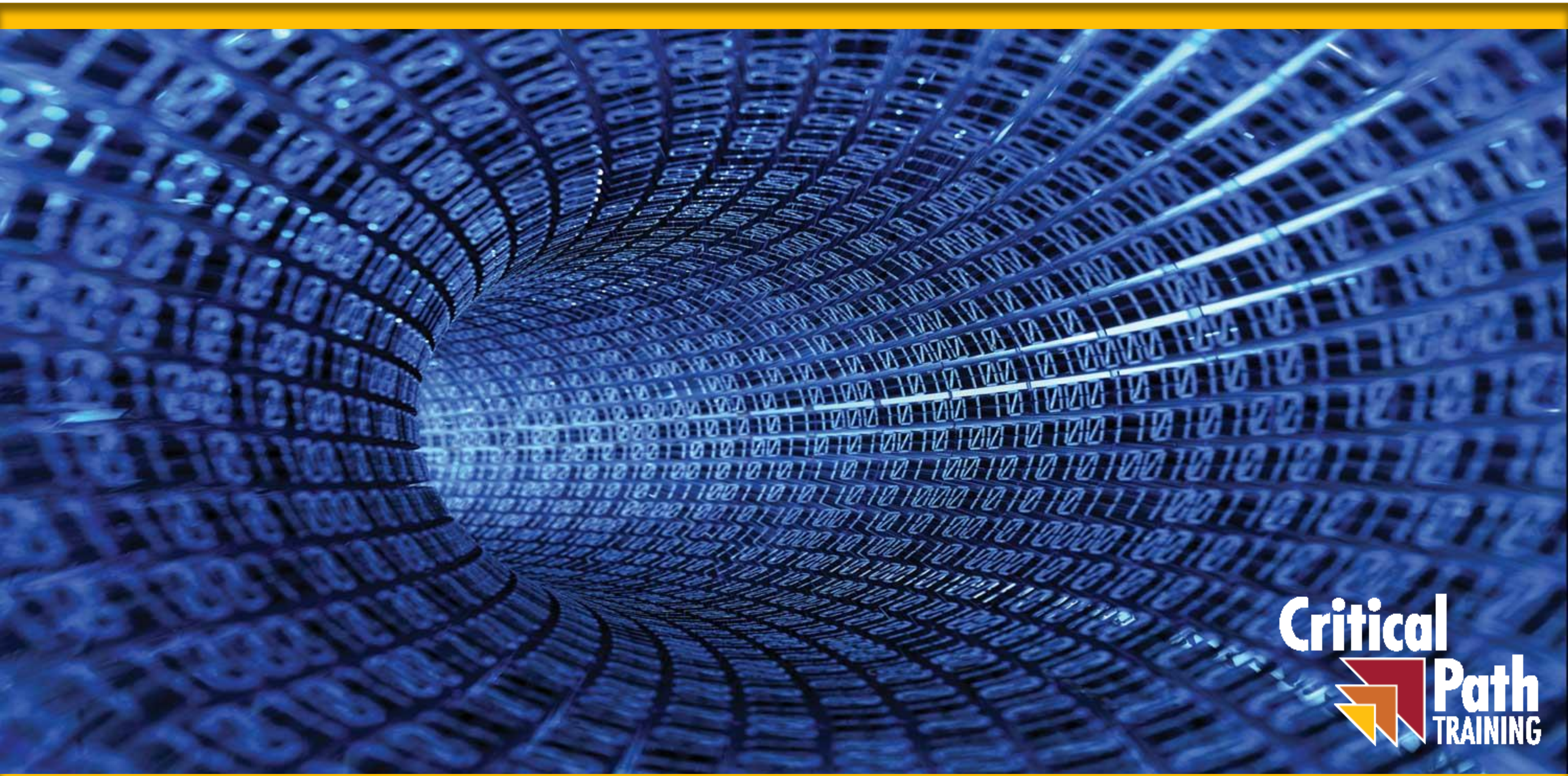
Developing Provider-hosted Add-ins



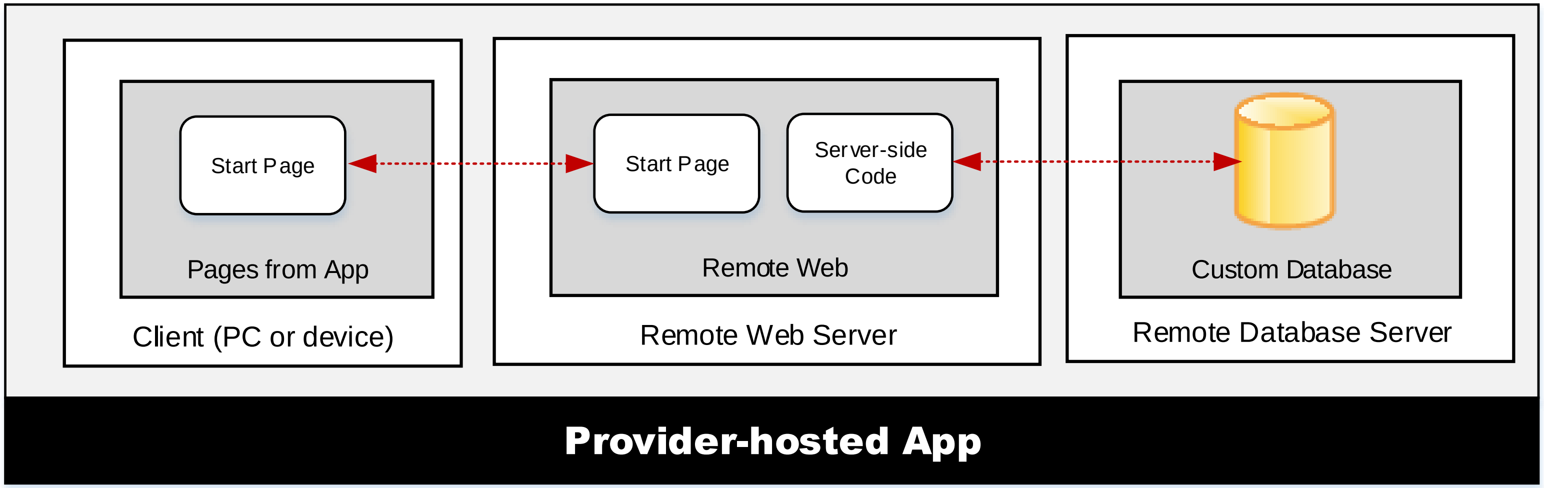
Critical Path TRAINING

Agenda

- Getting Started with Provider-hosted Apps
- User Interface Design for the Remote Web
- Working with ASP.NET MVC
- Creating Provider-hosted Apps using MVC5

Provider-Hosted App

- Developer responsible for deploying remote web
  - App deployed to remote web on remote web server
  - Developer deploys remote web prior to app installation
  - Developer often required to deploy database as well



Pros and Cons of Provider-hosted Apps

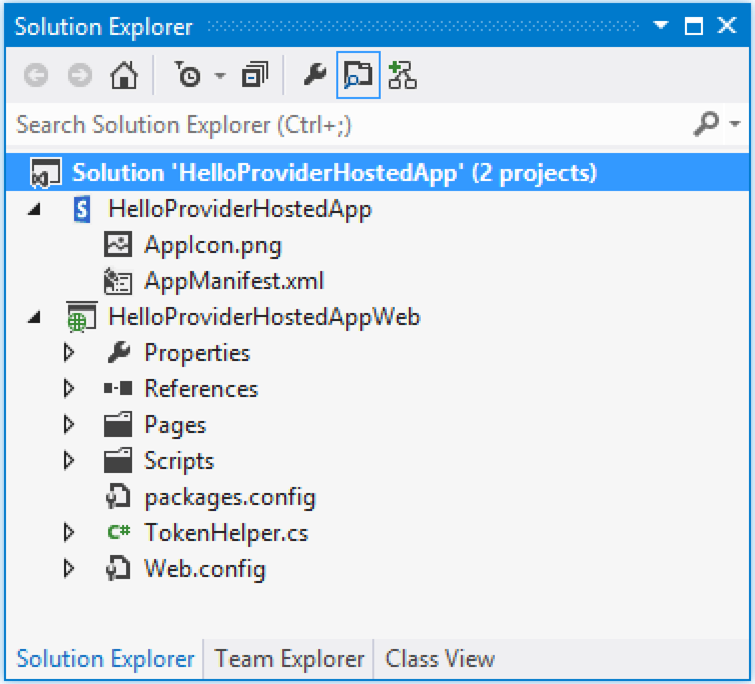
- Benefits of provider-hosted over SharePoint-hosted apps
  - You can write server-side .NET code using C# or VB.NET **([wahoo!])**
  - Your server-side code can access data in a custom database
  - Your server-side code has no cross-domain scripting constraints
  - You can leverage the support for remote event receivers
  - You can make CSOM/REST API calls using App-only permissions

- Negatives when compared to SharePoint-hosted apps
  - You must deal with hosting (i.e. deploy & manage) remote web
  - Requires extra code to acquire and manage security tokens

Provider-hosted App Projects

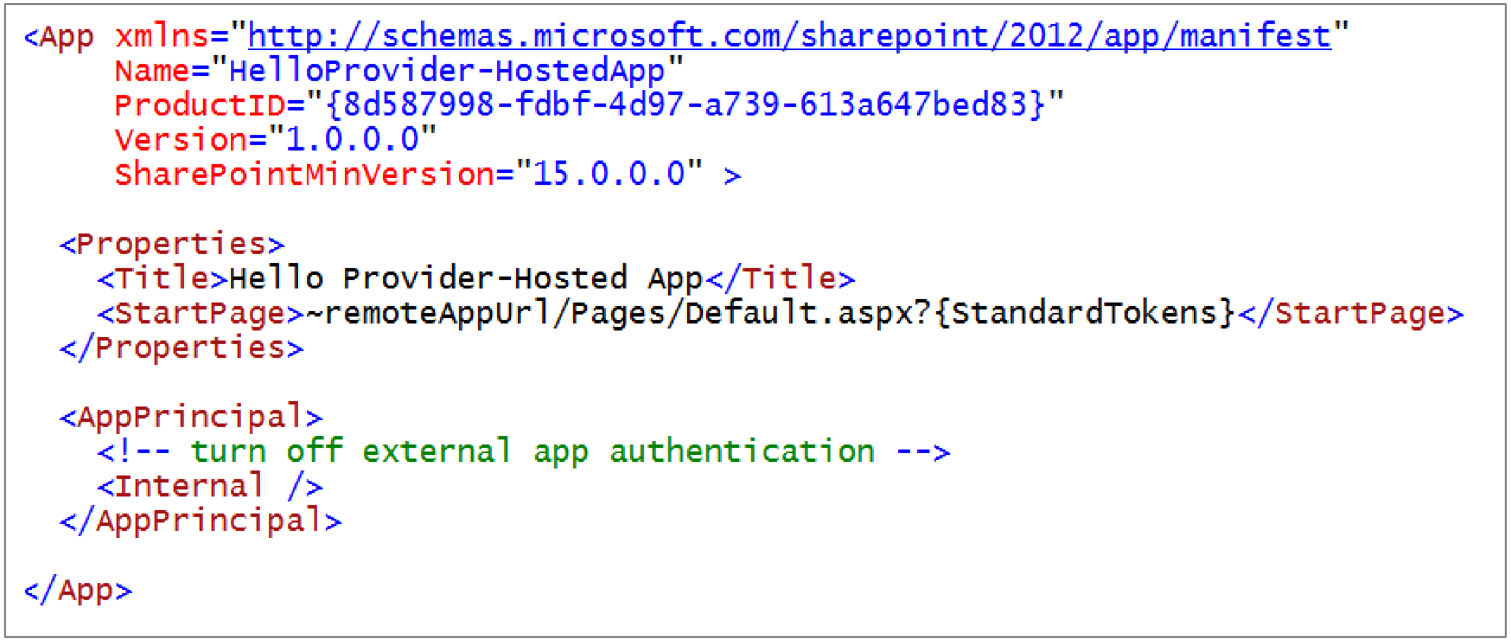
- Visual Studio create solution with two projects
  - SharePoint app project
  - ASP.NET Website project for remote web

    this project is known as the "web project"



AppManifest.xml

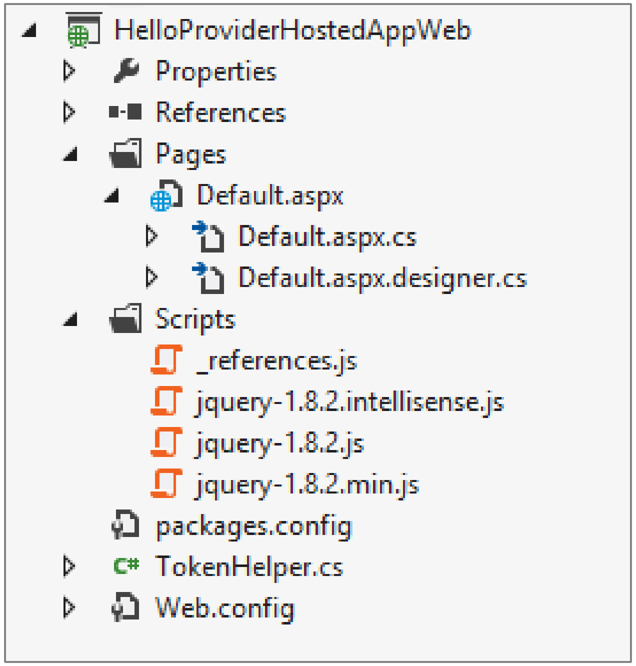
- Provider-hosted app adds requirements to App Manifest
  - StartPage must point to page in remote web
  - AppPrincipal requires app authentication settings
  - External app authentication can be disabled using Internal setting



```xml
<App xmlns="http://schemas.microsoft.com/sharepoint/2012/app/manifest"
     Name="HelloProvider-HostedApp"
     ProductID="{8d587998-fdbf-4d97-a739-613a647bed83}"
     Version="1.0.0.0"
     SharePointMinVersion="15.0.0.0" >

  <Properties>
    <Title>Hello Provider-Hosted App</Title>
    <StartPage>~remoteAppUrl/Pages/Default.aspx?{StandardTokens}</StartPage>
  </Properties>

  <AppPrincipal>
    <!-- turn off external app authentication -->
    <Internal />
  </AppPrincipal>

</App>
```

Implementing the Remote Web

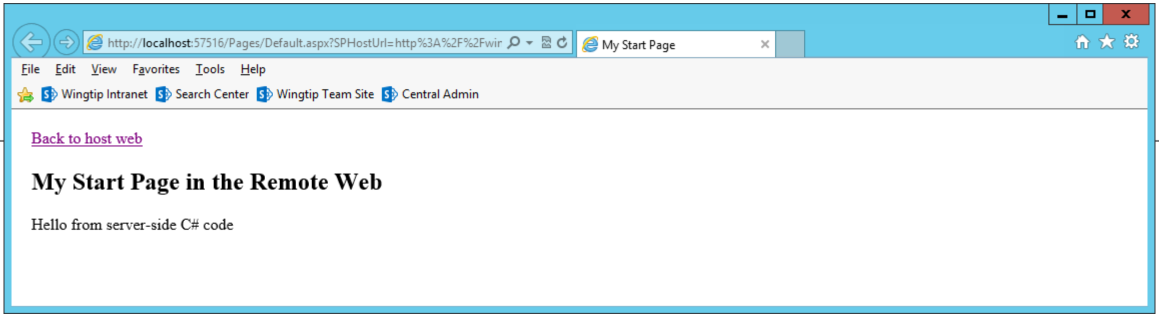
- Web project provides start point for remote web
  - web.config for site configuration of remote web
  - Default.aspx for start page with HTML layout
  - Default.aspx.cs for server-side C# code behind start page
  - Scripts folder with jQuery library already added
  - TokenHelper class for security-related programming



```
HelloProviderHostedAppWeb
    Properties
    References
    Pages
        Default.aspx
            Default.aspx.cs
            Default.aspx.designer.cs
    Scripts
        _references.js
        jquery-1.8.2.intellisense.js
        jquery-1.8.2.js
        jquery-1.8.2.min.js
    packages.config
    TokenHelper.cs
    Web.config
```

A Sample Start Page

```
<%@ Page Language="C#" AutoEventWireup="true"
         CodeBehind="Default.aspx.cs" Inherits="HelloProviderHostedAppWeb.Pages.Default" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">
    <title>My Start Page</title>
</head>

<body>
    <form id="form1" runat="server">
    <div>

      <!-- add HyperLink control to link back to host web-->
      <div><asp:HyperLink ID="linkHostWeb" runat="server">Back to host web</asp:HyperLink></div>

      <!-- add some HTML content to page -->
      <h2>My Start Page in the Remote Web</h2>

      <!-- -->
      <asp:PlaceHolder ID="PlaceHolderMain" runat="server"></asp:PlaceHolder>

    </div>
    </form>
</body>
</html>
```


http://localhost:57516/Pages/Default.aspx?SPHostUrl=http%3A%2F%2Fwir
My Start Page
File Edit View Favorites Tools Help
Wingtip Intranet
Search Center
Wingtip Team Site
Central Admin

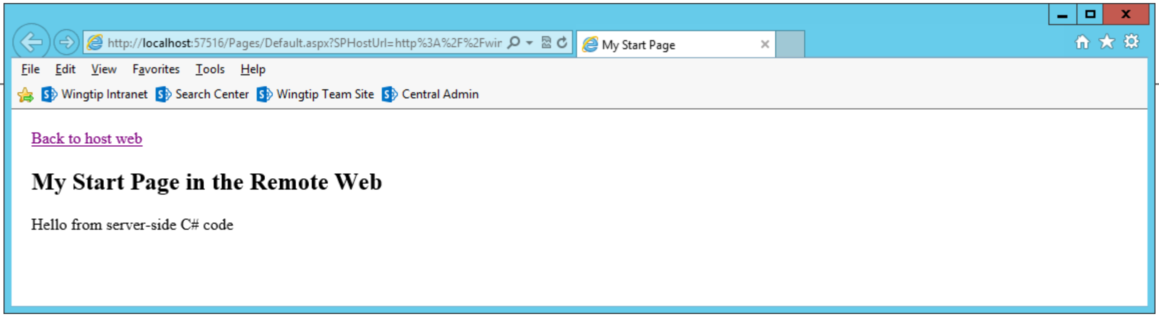
Back to host web

My Start Page in the Remote Web

Hello from server-side C# code

C# Code Behind Sample Start Page

```
namespace HelloProviderHostedAppWeb.Pages {
  public partial class Default : System.Web.UI.Page {

    protected void Page_Load(object sender, EventArgs e) {

      // delete all existing code added by Visual Studio - it requires authentication

      // Configure ASP.NET Hyperlink control with value from SPHostUrl querystring
      linkHostWeb.NavigateUrl = Request.QueryString["SPHostUrl"];

      // add some content to the page using server-side code
      PlaceHolderMain.Controls.Add( new LiteralControl("Hello from server-side C# code"));

    }
  }
}
```


http://localhost:57516/Pages/Default.aspx?SPHostUrl=http%3A%2F%2Fwir
My Start Page
File Edit View Favorites Tools Help
Wingtip Intranet
Search Center
Wingtip Team Site
Central Admin

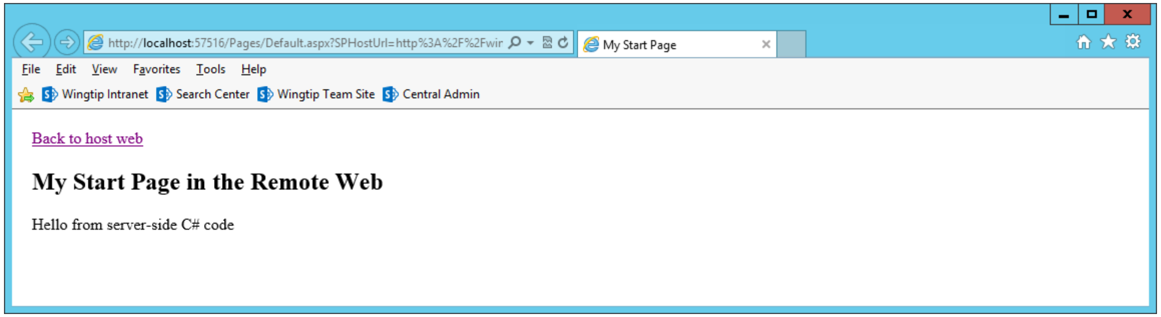
Back to host web

My Start Page in the Remote Web

Hello from server-side C# code

Debugging the Remote Web in IIS Express

- Visual Studio debugging involves IIS Express
  - URL created in **localhost** domain (e.g. **https://localhost:57516**)
  - Port number for Remote Web project selected automatically behind scenes



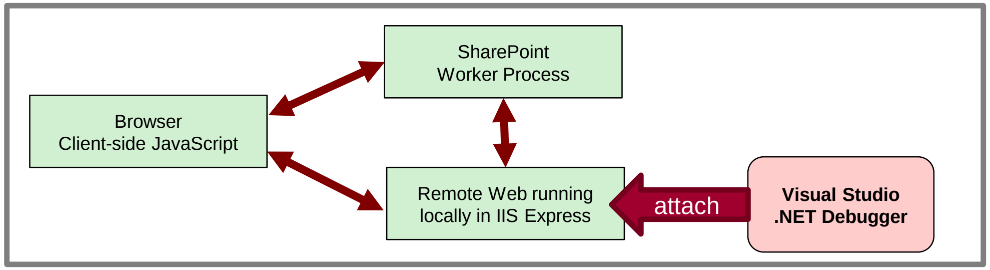
http://localhost:57516/Pages/Default.aspx?SPHostUrl=http%3A%2F%2Fwir
My Start Page
File Edit View Favorites Tools Help
Wingtip Intranet
Search Center
Wingtip Team Site
Central Admin

Back to host web

My Start Page in the Remote Web

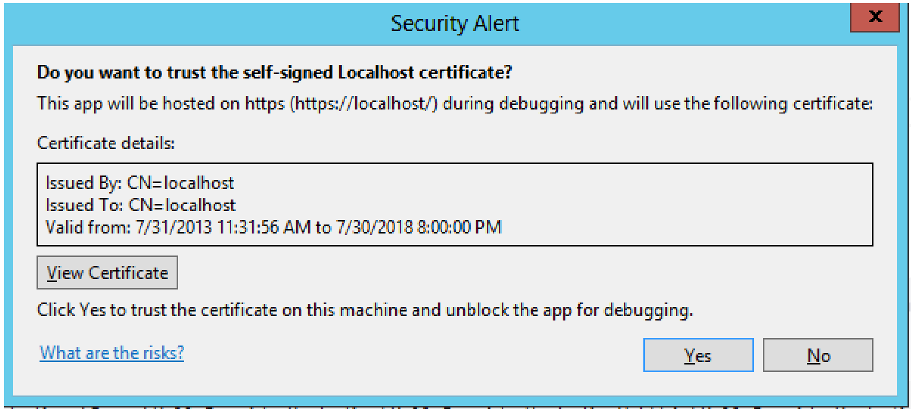
Hello from server-side C# code

- IIS Express debugging support allows you to debug…
  - Server-side C# code behind pages in the remote web
  - Web service calls from browser to remote web
  - Web service calls from SharePoint to remote web



Debugging Code in the Remote Web

- Remote Web can optionally use SSL
  - Pages can be served using HTTPS or HTTP
  - Use of SSL is usually preferred
  - IIS Express can configure SSL through https://localhost
  - Visual Studio registers self-signed certificate on first use



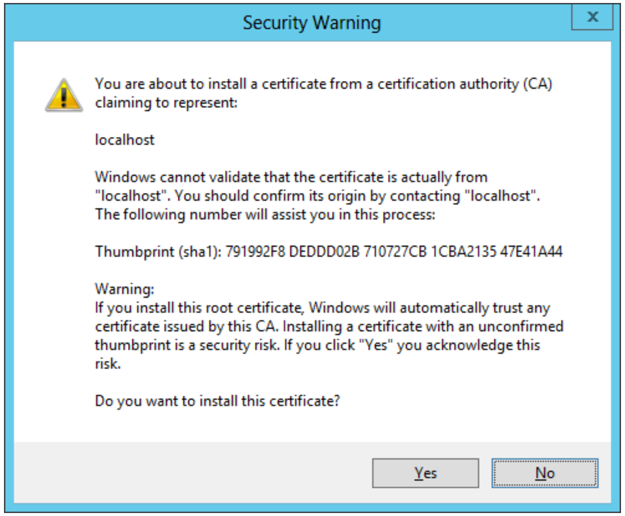
Do you want to trust the self-signed Localhost certificate?

This app will be hosted on https (https://localhost/) during debugging and will use the following certificate:

Certificate details:

Issued By: CN=localhost
Issued To: CN=localhost
Valid from: 7/31/2013 11:31:56 AM to 7/30/2018 8:00:00 PM

Click Yes to trust the certificate on this machine and unblock the app for debugging.



You are about to install a certificate from a certification authority (CA) claiming to represent:

localhost

Windows cannot validate that the certificate is actually from "localhost". You should confirm its origin by contacting "localhost". The following number will assist you in this process:

Thumbprint (sha1): 791992F8 DEDDD02B 710727CB 1CBA2135 47E41A44

Warning:
If you install this root certificate, Windows will automatically trust any certificate issued by this CA. Installing a certificate with an unconfirmed thumbprint is a security risk. If you click "Yes" you acknowledge this risk.

Do you want to install this certificate?

DEMO

Creating the ‘Hello World’ Provider-Hosted App

Agenda

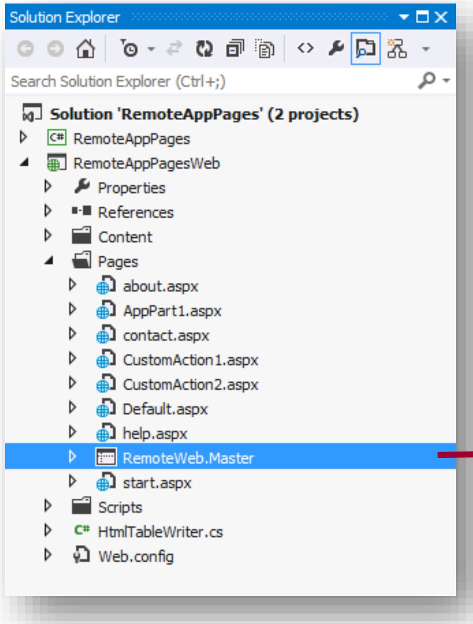
- ✓ Getting Started with Provider-hosted Apps
- ➢ User Interface Design for the Remote Web
- Working with ASP.NET MVC
- Creating Provider-hosted Apps using MVC5

Designing the Remote Web User Interface

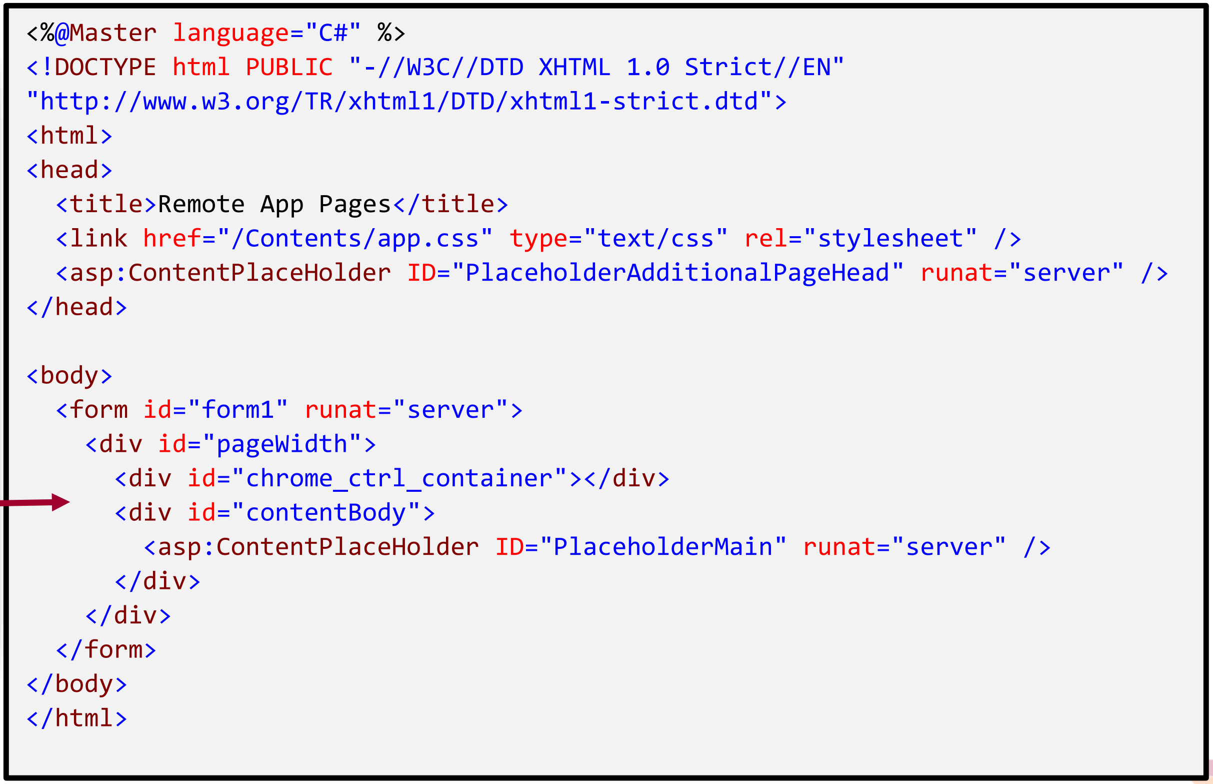
- Create user interface using single page app pattern
  - Start page serves as exclusive user entry point into remote web
  - Start page designed with HTML markup and ASP.NET controls
  - Start page can be written with server-side C# code
  - Start page can also include client-side JavaScript code
  - JavaScript code can call to custom web services in remote web
- Create user interface as a multi-page app
  - Users can navigate from start page to other pages in remote web
  - ASP.NET master page can create consistent UI across pages
  - Master page can leverage SharePoint Chrome control
  - Multi-page app loses major benefits of single-page app pattern
  - Extra work required to track start page POST data across requests

Using Master Pages in the Remote Web

- Pages in remote web live outside SharePoint
  - You have responsibility and freedom to build entire UI
  - Often makes sense to create remote web master page



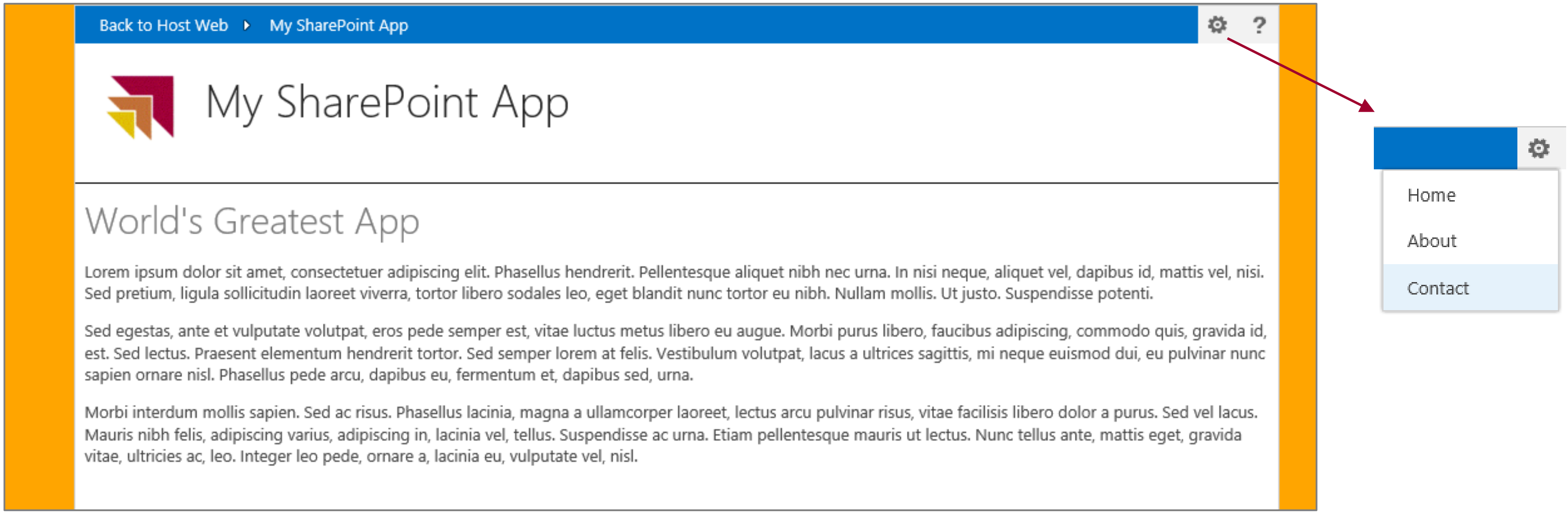
- Solution 'RemoteAppPages' (2 projects)
  - RemoteAppPages
  - RemoteAppPagesWeb
    - Properties
    - References
    - Content
    - Pages
      - about.aspx
      - AppPart1.aspx
      - contact.aspx
      - CustomAction1.aspx
      - CustomAction2.aspx
      - Default.aspx
      - help.aspx
      - RemoteWeb.Master
      - start.aspx
    - Scripts
    - HtmlTableWriter.cs
    - Web.config



```
<%@Master language="C#" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Remote App Pages</title>
  <link href="/Contents/app.css" type="text/css" rel="stylesheet" />
  <asp:ContentPlaceHolder ID="PlaceholderAdditionalPageHead" runat="server" />
</head>

<body>
  <form id="form1" runat="server">
    <div id="pageWidth">
      <div id="chrome_ctrl_container"></div>
      <div id="contentBody">
        <asp:ContentPlaceHolder ID="PlaceholderMain" runat="server" />
      </div>
    </div>
  </form>
</body>
</html>
```

Leveraging the Chrome Control

- Optional JavaScript component for use in Remote Web
  - Used to create top banner of page with SharePoint look and feel
  - Provides link to host web, configurable navigation and help link
  - Can pull custom styles used in host web



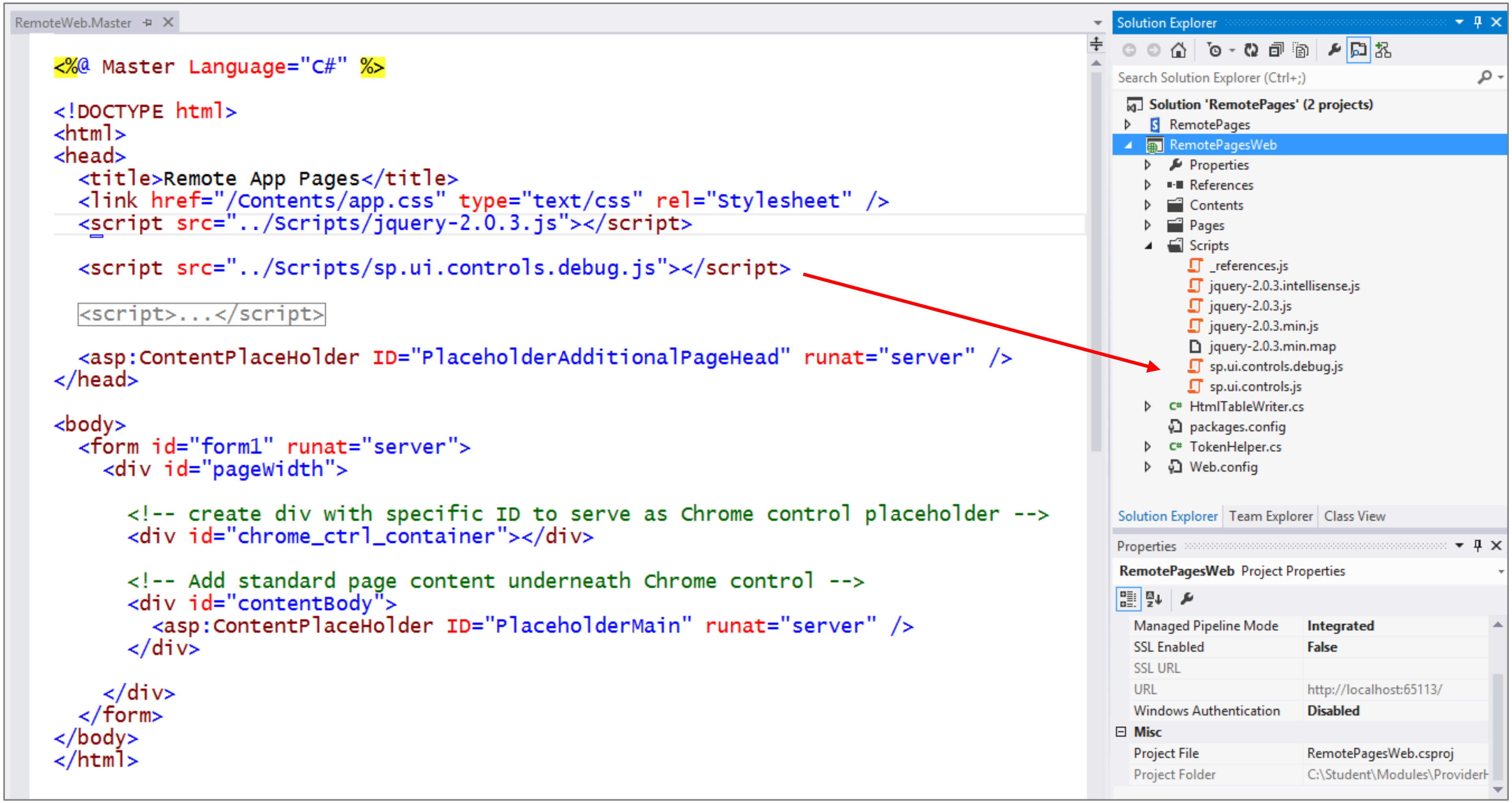
World's Greatest App

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Phasellus hendrerit. Pellentesque aliquet nibh nec urna. In nisi neque, aliquet vel, dapibus id, mattis vel, nisi. Sed pretium, ligula sollicitudin laoreet viverra, tortor libero sodales leo, eget blandit nunc tortor eu nibh. Nullam mollis. Ut justo. Suspendisse potenti.

Sed egestas, ante et vulputate volutpat, eros pede semper est, vitae luctus metus libero eu augue. Morbi purus libero, faucibus adipiscing, commodo quis, gravida id, est. Sed lectus. Praesent elementum hendrerit tortor. Sed semper lorem at felis. Vestibulum volutpat, lacus a ultrices sagittis, mi neque euismod dui, eu pulvinar nunc sapien ornare nisl. Phasellus pede arcu, dapibus eu, fermentum et, dapibus sed, urna.

Morbi interdum mollis sapien. Sed ac risus. Phasellus lacinia, magna a ullamcorper laoreet, lectus arcu pulvinar risus, vitae facilisis libero dolor a purus. Sed vel lacus. Mauris nibh felis, adipiscing varius, adipiscing in, lacinia vel, tellus. Suspendisse ac urna. Etiam pellentesque mauris ut lectus. Nunc tellus ante, mattis eget, gravida vitae, ultricies ac, leo. Integer leo pede, ornare a, lacinia eu, vulputate vel, nisl.

Using the Chrome control

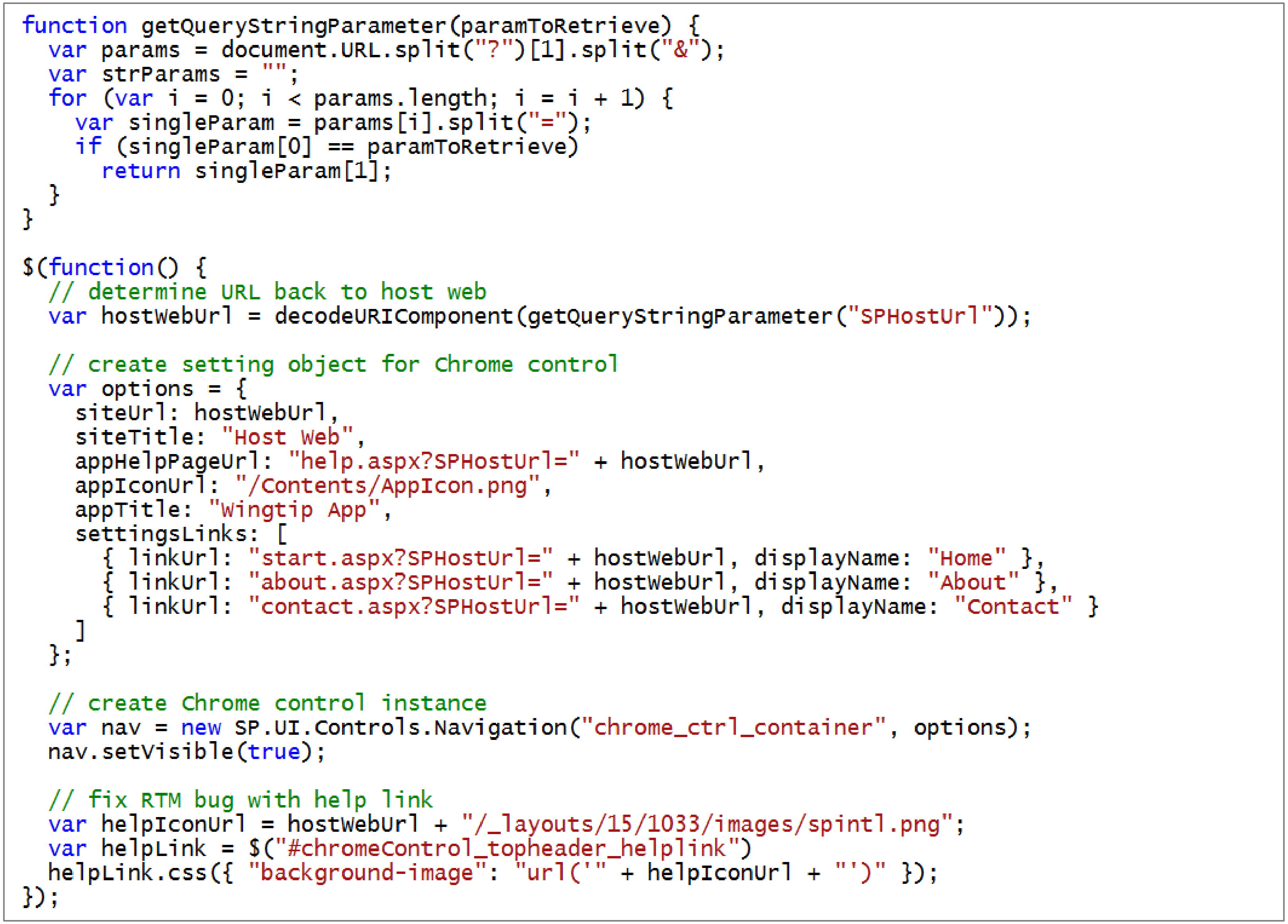


```
<%@ Master Language="C#" %>

<!DOCTYPE html>
<html>
<head>
  <title>Remote App Pages</title>
  <link href="/Contents/app.css" type="text/css" rel="Stylesheet" />
  <script src="../Scripts/jquery-2.0.3.js"></script>

  <script src="../Scripts/sp.ui.controls.debug.js"></script>

  <script>...</script>

  <asp:ContentPlaceHolder ID="PlaceholderAdditionalPageHead" runat="server" />
</head>

<body>
  <form id="form1" runat="server">
    <div id="pageWidth">

      <!-- create div with specific ID to serve as Chrome control placeholder -->
      <div id="chrome_ctrl_container"></div>

      <!-- Add standard page content underneath Chrome control -->
      <div id="contentBody">
        <asp:ContentPlaceHolder ID="PlaceholderMain" runat="server" />
      </div>

    </div>
  </form>
</body>
</html>
```

RemotePagesWeb Project Properties

Managed Pipeline Mode	**Integrated**
SSL Enabled	**False**
SSL URL	
URL	http://localhost:65113/
Windows Authentication	**Disabled**
Misc	
Project File	RemotePagesWeb.csproj
Project Folder	C:\Student\Modules\ProviderH

Initializing the Chrome Control



```
function getQueryStringParameter(paramToRetrieve) {
  var params = document.URL.split("?")[1].split("&");
  var strParams = "";
  for (var i = 0; i < params.length; i = i + 1) {
    var singleParam = params[i].split("=");
    if (singleParam[0] == paramToRetrieve)
      return singleParam[1];
  }
}

$(function() {
  // determine URL back to host web
  var hostWebUrl = decodeURIComponent(getQueryStringParameter("SPHostUrl"));

  // create setting object for Chrome control
  var options = {
    siteUrl: hostWebUrl,
    siteTitle: "Host Web",
    appHelpPageUrl: "help.aspx?SPHostUrl=" + hostWebUrl,
    appIconUrl: "/Contents/AppIcon.png",
    appTitle: "Wingtip App",
    settingsLinks: [
      { linkUrl: "start.aspx?SPHostUrl=" + hostWebUrl, displayName: "Home" },
      { linkUrl: "about.aspx?SPHostUrl=" + hostWebUrl, displayName: "About" },
      { linkUrl: "contact.aspx?SPHostUrl=" + hostWebUrl, displayName: "Contact" }
    ]
  };

  // create Chrome control instance
  var nav = new SP.UI.Controls.Navigation("chrome_ctrl_container", options);
  nav.setVisible(true);

  // fix RTM bug with help link
  var helpIconUrl = hostWebUrl + "/_layouts/15/1033/images/spintl.png";
  var helpLink = $("#chromeControl_topheader_helplink")
  helpLink.css({ "background-image": "url('" + helpIconUrl + "')" });
});
```

DEMO

Creating Pages in the Remote Web using the Chrome Control

Agenda

✓ Getting Started with Provider-hosted Apps

✓ User Interface Design for the Remote Web

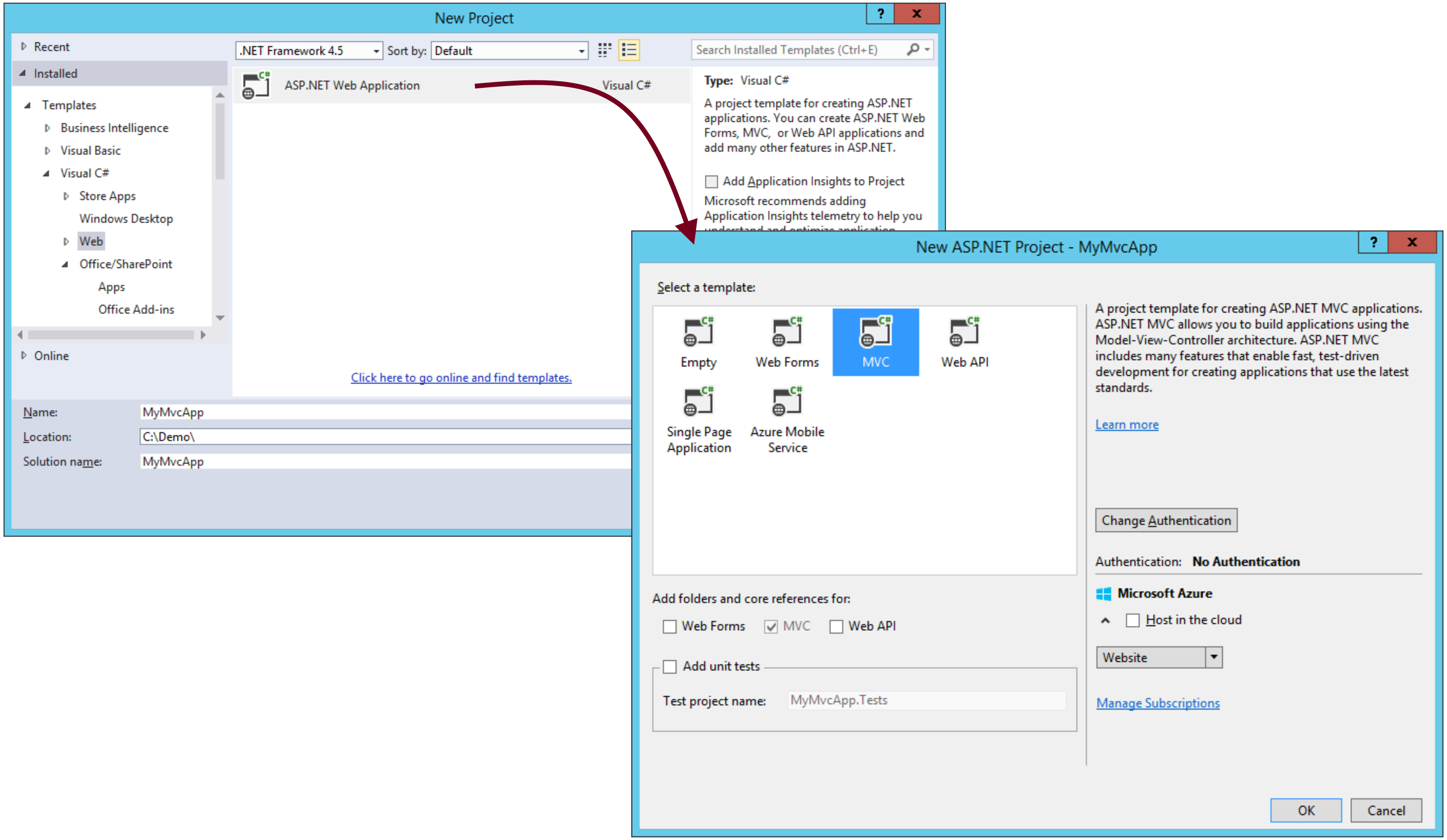
➢ Working with ASP.NET MVC

- Creating Provider-hosted Apps using MVC5

Web Forms Versus MVC

- ASP.NET provides two different platforms
  - ASP.NET Web Forms (e.g. ASPX files)
  - ASP.NET MVC
- MVC provides better platform for the web
  - More flexible routing
  - Lighter-weight
  - Richer templating
  - Better C# integration
  - Unit testing

Creating an MVC Project



.NET Framework 4.5

Sort by: Default

Type: Visual C#

A project template for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

□ Add Application Insights to Project

Microsoft recommends adding Application Insights telemetry to help you understand and optimize application

Name: MyMvcApp

Location: C:\Demo\

Solution name: MyMvcApp

Select a template:

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Authentication: **No Authentication**

Microsoft Azure

□ Host in the cloud

Website

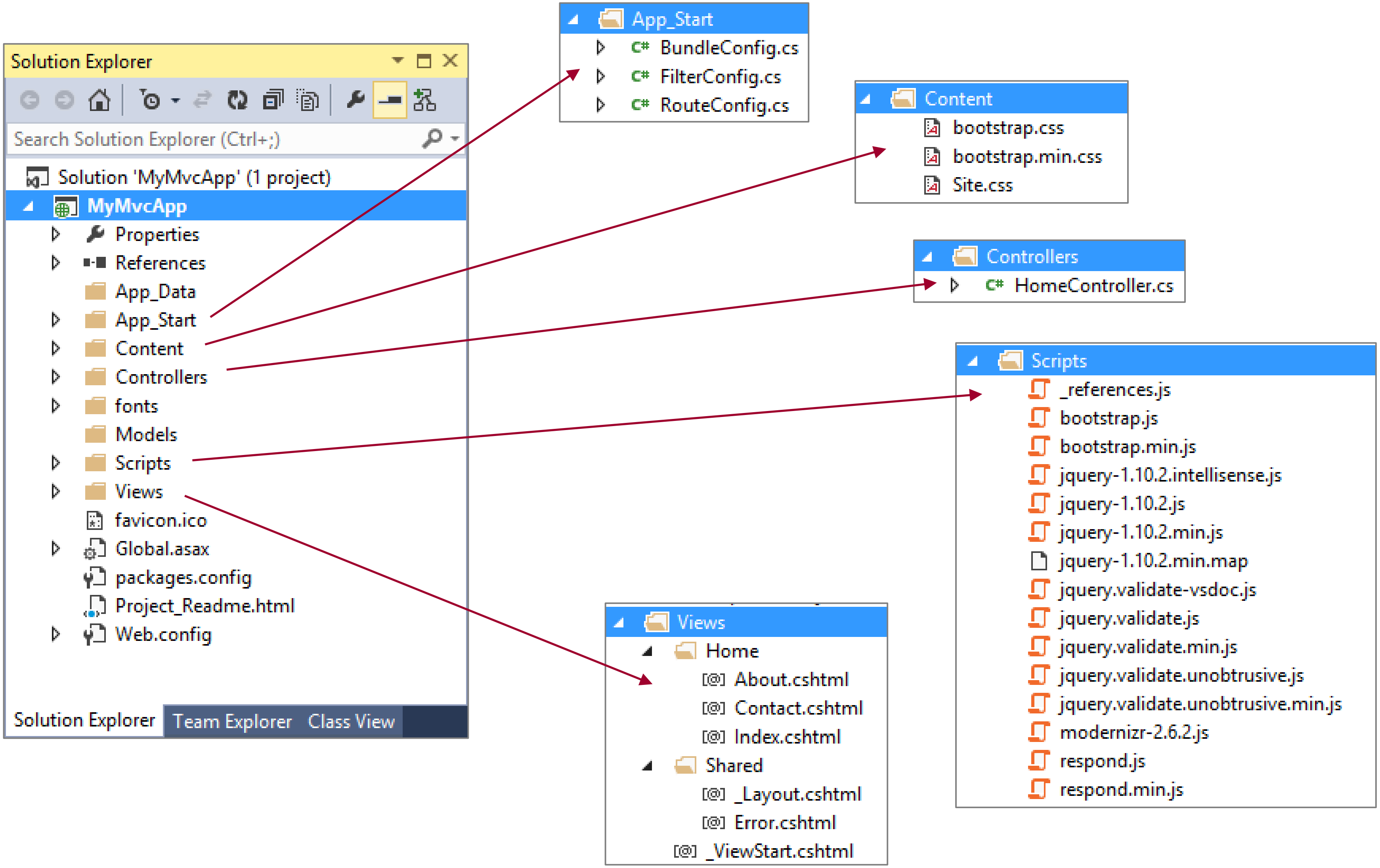
Add folders and core references for:

□ Web Forms ☑ MVC □ Web API

□ Add unit tests

Test project name: MyMvcApp.Tests

MVC App Project Files



Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'MyMvcApp' (1 project)
  - MyMvcApp
    - Properties
    - References
    - App_Data
    - App_Start
    - Content
    - Controllers
    - fonts
    - Models
    - Scripts
    - Views
    - favicon.ico
    - Global.asax
    - packages.config
    - Project_Readme.html
    - Web.config

Solution Explorer | Team Explorer | Class View

- App_Start
  - BundleConfig.cs
  - FilterConfig.cs
  - RouteConfig.cs

- Content
  - bootstrap.css
  - bootstrap.min.css
  - Site.css

- Controllers
  - HomeController.cs

- Scripts
  - _references.js
  - bootstrap.js
  - bootstrap.min.js
  - jquery-1.10.2.intellisense.js
  - jquery-1.10.2.js
  - jquery-1.10.2.min.js
  - jquery-1.10.2.min.map
  - jquery.validate-vsdoc.js
  - jquery.validate.js
  - jquery.validate.min.js
  - jquery.validate.unobtrusive.js
  - jquery.validate.unobtrusive.min.js
  - modernizr-2.6.2.js
  - respond.js
  - respond.min.js

- Views
  - Home
    - About.cshtml
    - Contact.cshtml
    - Index.cshtml
  - Shared
    - _Layout.cshtml
    - Error.cshtml
  - _ViewStart.cshtml

DEMO

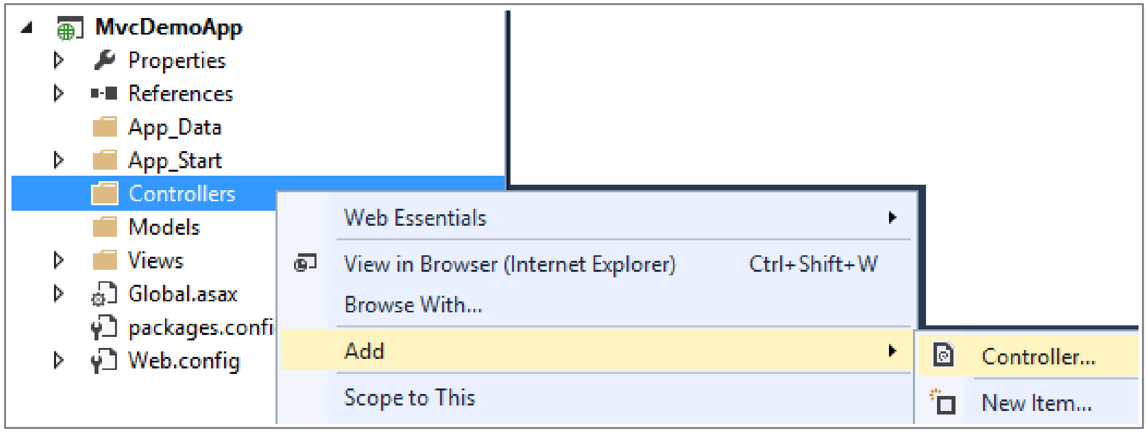
Creating a Simple App with ASP.NET MVC

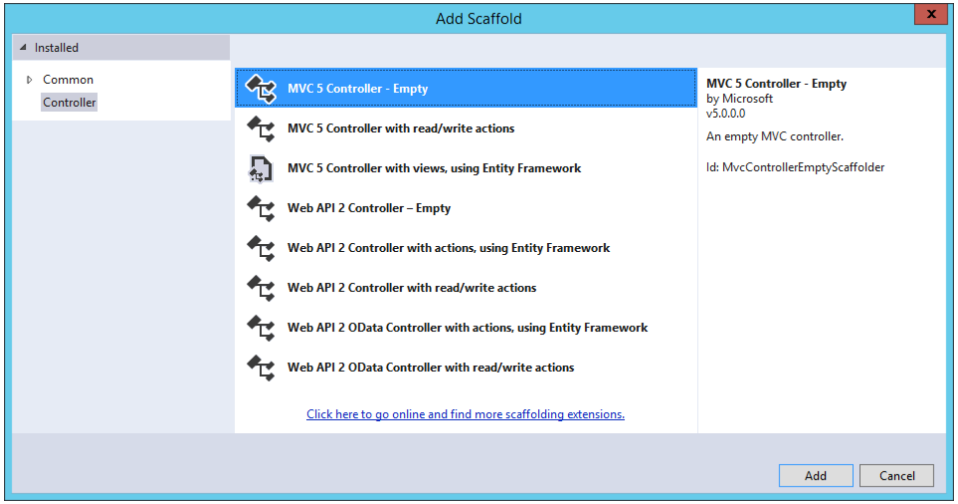
Understanding Controllers

- A set of classes that manage…
  - processing incoming HTTP requests
  - communication to and from user
  - overall application flow and application-specific logic

- Every controller has one or more Actions
  - It’s critical to understand the role of Actions in MVC

Adding a Controller

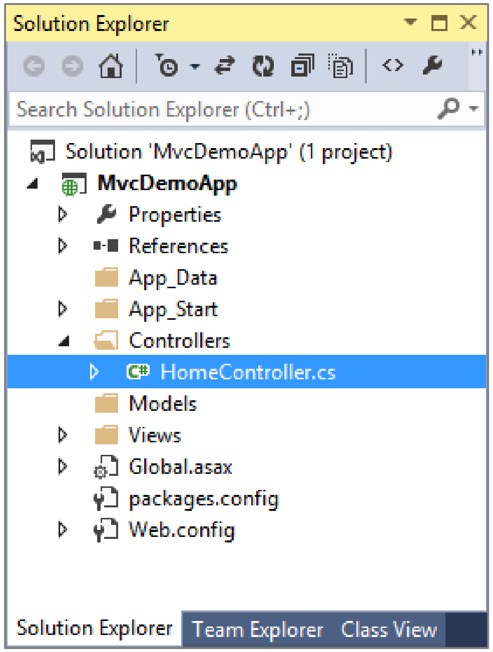



Add Scaffold
Installed
Common
Controller
MVC 5 Controller - Empty
MVC 5 Controller with read/write actions
MVC 5 Controller with views, using Entity Framework
Web API 2 Controller – Empty
Web API 2 Controller with actions, using Entity Framework
Web API 2 Controller with read/write actions
Web API 2 OData Controller with actions, using Entity Framework
Web API 2 OData Controller with read/write actions
Click here to go online and find more scaffolding extensions.
Add
Cancel

MVC 5 Controller - Empty
by Microsoft
v5.0.0.0

An empty MVC controller.

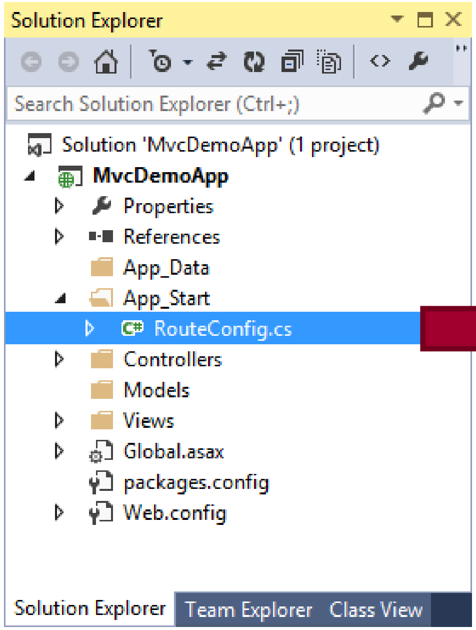
Id: MvcControllerEmptyScaffolder

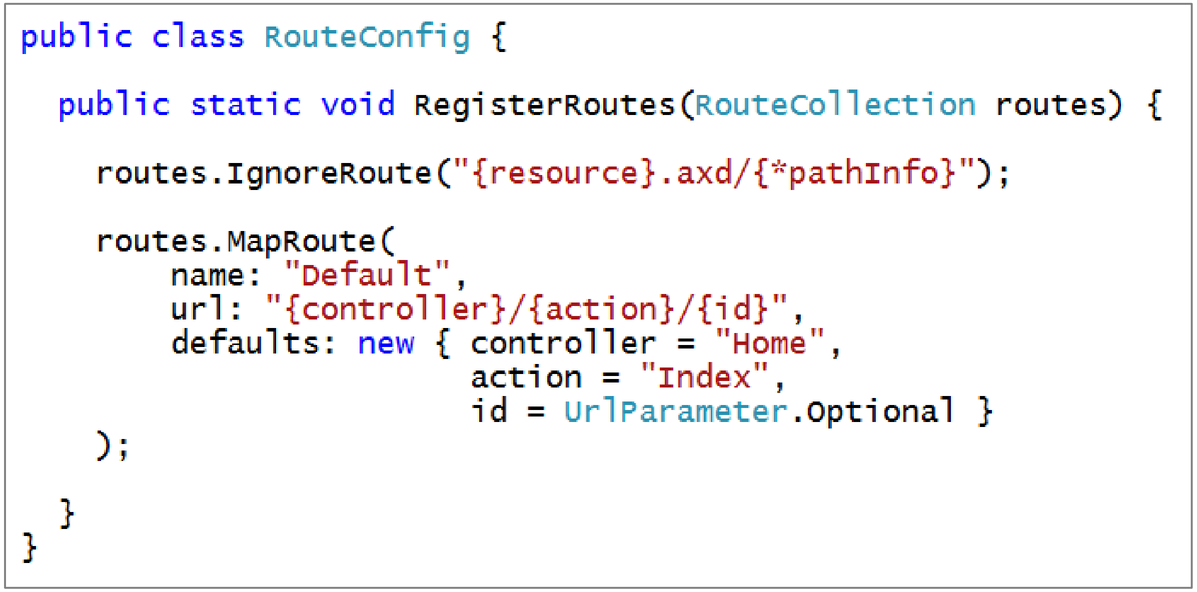


Wiring Up a Controller

- When you ad a controller…
  - Visual Studio updates **RouteConfig** class
  - Routing scheme defined using standard format

    `{controller}/{action}/{id}`

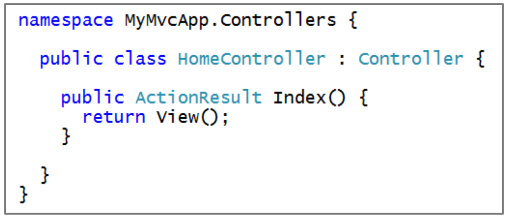




```
public class RouteConfig {

  public static void RegisterRoutes(RouteCollection routes) {

    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

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

  }
}
```

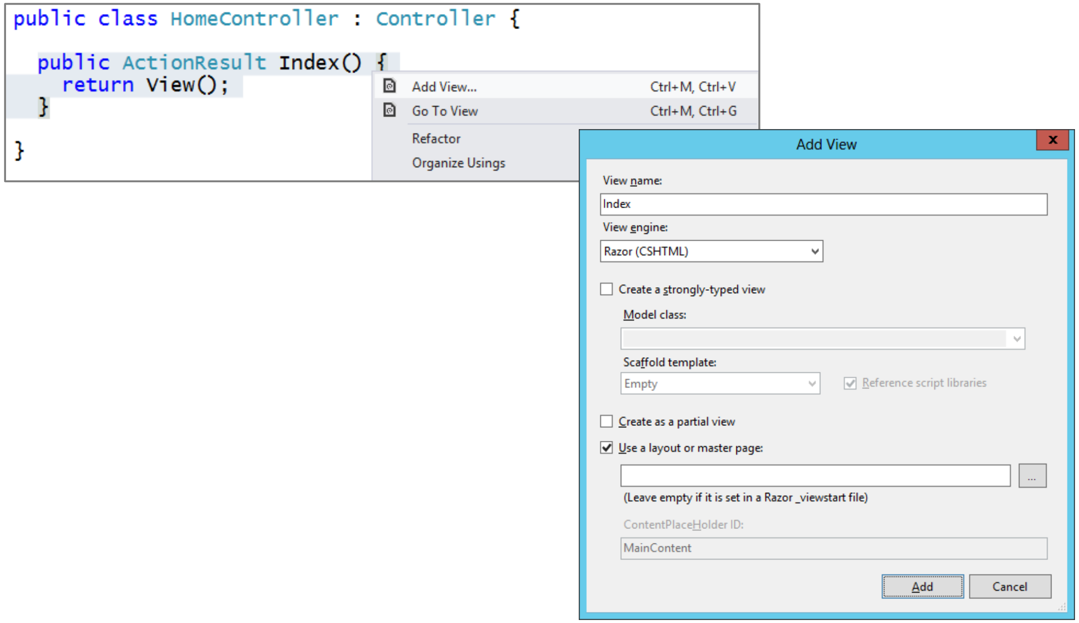
Creating a View from a Controller Action

- Controller method often return **ActionResult**



```
namespace MyMvcApp.Controllers {
  public class HomeController : Controller {
    public ActionResult Index() {
      return View();
    }
  }
}
```

- Right-click on Controller method to generate its view



```
public class HomeController : Controller {
  public ActionResult Index() {
    return View();
  }
}
```

View name: Index

View engine: Razor (CSHTML)

□ Create a strongly-typed view

Model class:

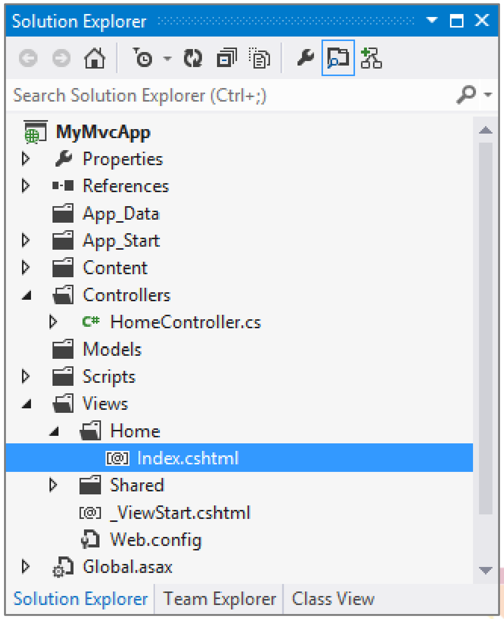
Scaffold template: Empty ☑ Reference script libraries

□ Create as a partial view

☑ Use a layout or master page:

(Leave empty if it is set in a Razor _viewstart file)

ContentPlaceHolder ID: MainContent



MyMvcApp
Properties
References
App_Data
App_Start
Content
Controllers
HomeController.cs
Models
Scripts
Views
Home
Index.cshtml
Shared
_ViewStart.cshtml
Web.config
Global.asax

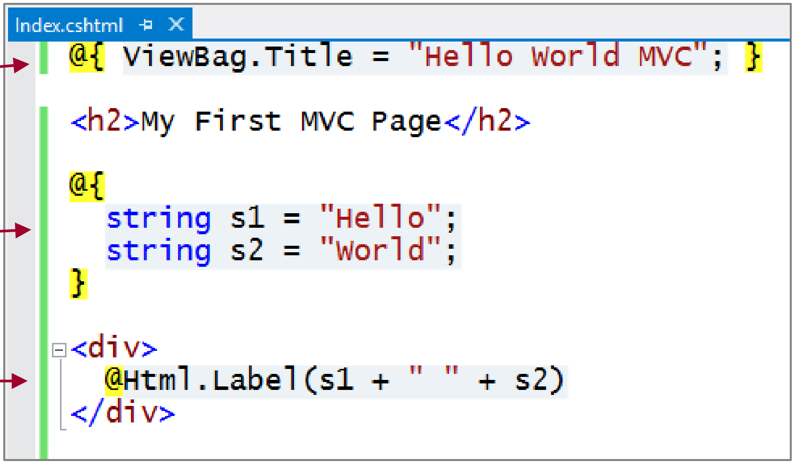
Customizing the View

- Views are creating using the Razor engine
  - Provides a lean and elegant way to create HTML pages

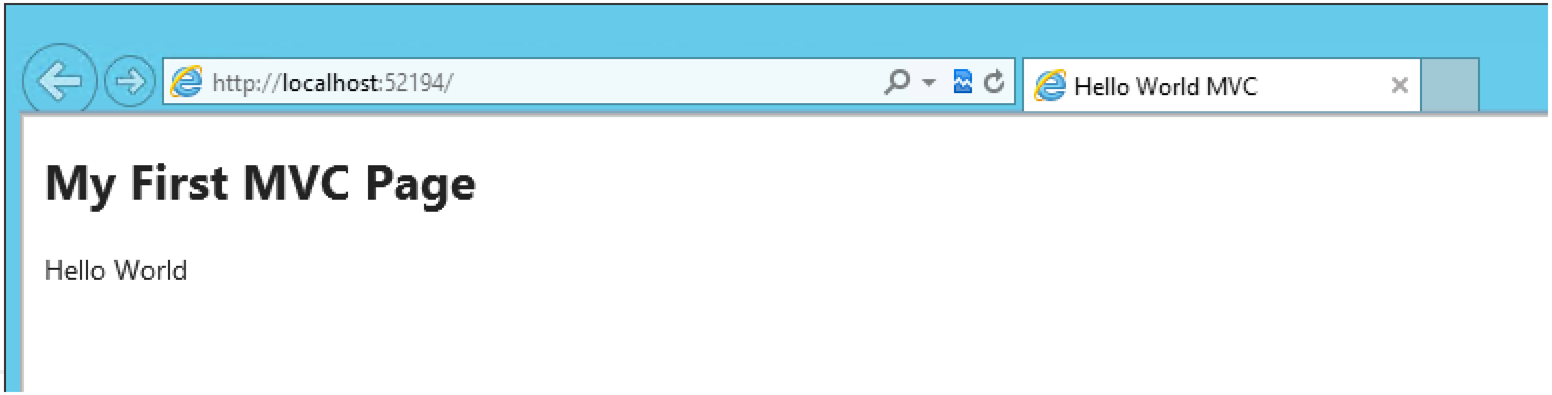
Use data passed from controller

Write in-line C# code

Construct pages using HTML Helpers



```
@{ ViewBag.Title = "Hello World MVC"; }

<h2>My First MVC Page</h2>

@{
    string s1 = "Hello";
    string s2 = "World";
}

<div>
    @Html.Label(s1 + " " + s2)
</div>
```

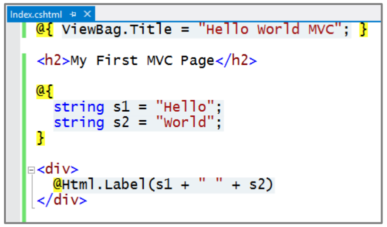


My First MVC Page

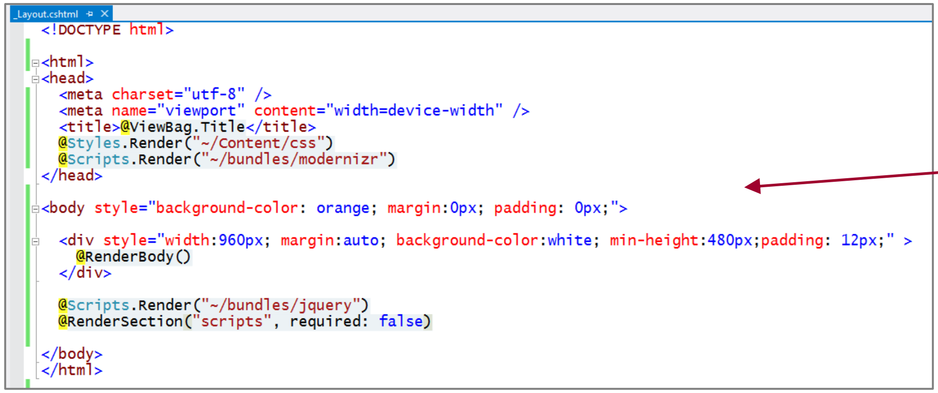
Hello World

Customizing the Shared View

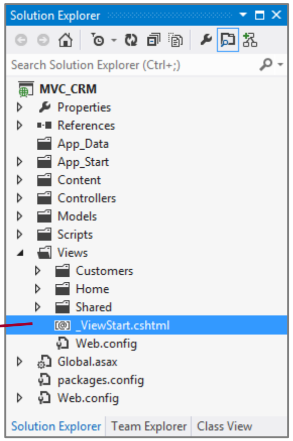
- MVC provides Shared Views
  - Provides same purpose as master pages in ASP.NET web forms
  - Default MVC shared view is named **_ViewStart.cshtml**

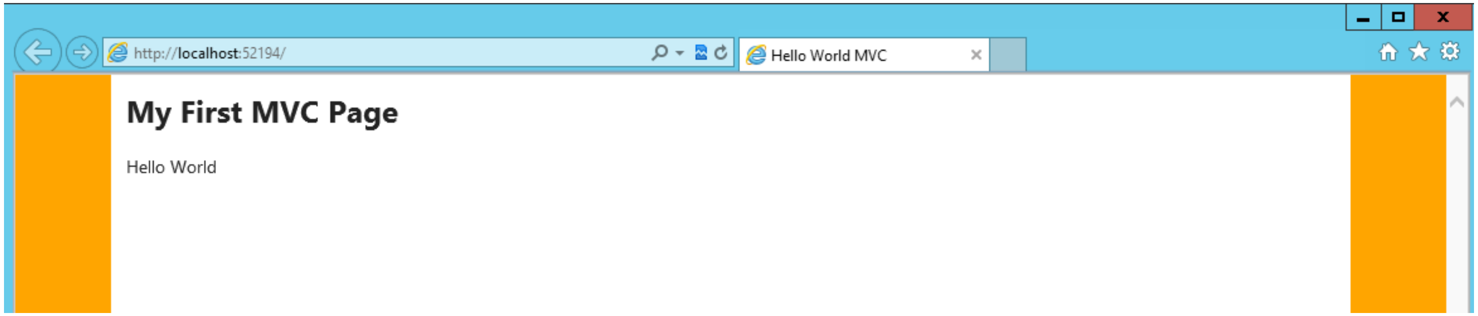


```
@{ ViewBag.Title = "Hello World MVC"; }

<h2>My First MVC Page</h2>

@{
   string s1 = "Hello";
   string s2 = "World";
}

<div>
   @Html.Label(s1 + " " + s2)
</div>
```



```
<!DOCTYPE html>

<html>
<head>
   <meta charset="utf-8" />
   <meta name="viewport" content="width=device-width" />
   <title>@ViewBag.Title</title>
   @Styles.Render("~/Content/css")
   @Scripts.Render("~/bundles/modernizr")
</head>

<body style="background-color: orange; margin:0px; padding: 0px;">

   <div style="width:960px; margin:auto; background-color:white; min-height:480px;padding: 12px;" >
     @RenderBody()
   </div>

   @Scripts.Render("~/bundles/jquery")
   @RenderSection("scripts", required: false)

</body>
</html>
```





My First MVC Page

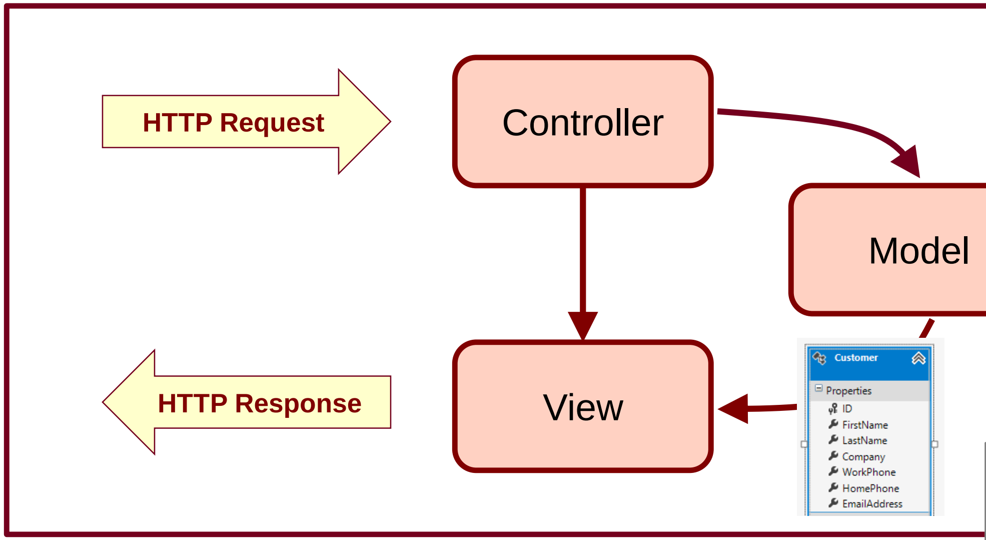
Hello World

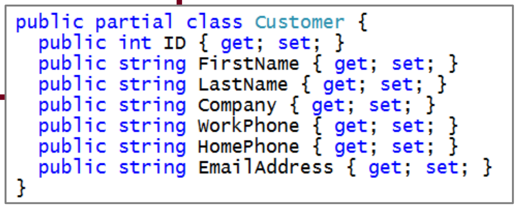
DEMO

Working with Controllers and Views

Motivation for Strongly-typed Models

- MVC designed based on strongly-typed models
  - Controller creates model object and passes it to view
  - Razor view engine supplies IntelliSense for model behind view
  - HTML helpers make it easy to create views and forms





```
public partial class Customer {
  public int ID { get; set; }
  public string FirstName { get; set; }
  public string LastName { get; set; }
  public string Company { get; set; }
  public string WorkPhone { get; set; }
  public string HomePhone { get; set; }
  public string EmailAddress { get; set; }
}
```

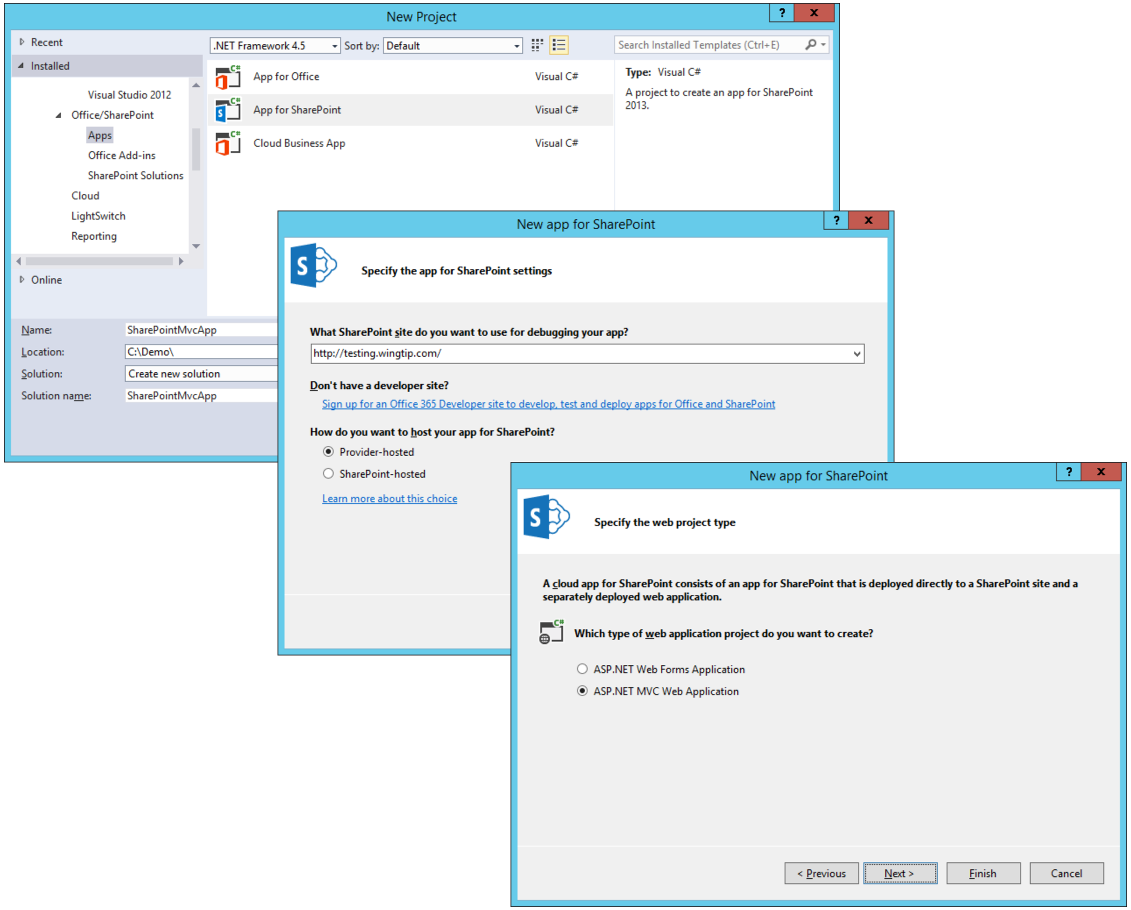
DEMO

Using a Strongly-typed Model

Agenda

- ✓ Getting Started with Provider-hosted Apps
- ✓ User Interface Design for the Remote Web
- ✓ Working with ASP.NET MVC
- ➢ Creating Provider-hosted Apps using MVC5

Creating a Provider-hosted App with MVC



New Project

- Recent
- Installed
  - Visual Studio 2012
  - Office/SharePoint
    - Apps
    - Office Add-ins
    - SharePoint Solutions
  - Cloud
  - LightSwitch
  - Reporting
- Online

.NET Framework 4.5 · Sort by: Default · Search Installed Templates (Ctrl+E)

App for Office	Visual C#
App for SharePoint	Visual C#
Cloud Business App	Visual C#

Type: Visual C#

A project to create an app for SharePoint 2013.

Name: SharePointMvcApp

Location: C:\Demo\

Solution: Create new solution

Solution name: SharePointMvcApp

New app for SharePoint

Specify the app for SharePoint settings

What SharePoint site do you want to use for debugging your app?

http://testing.wingtip.com/

Don't have a developer site?

Sign up for an Office 365 Developer site to develop, test and deploy apps for Office and SharePoint

How do you want to host your app for SharePoint?

● Provider-hosted

○ SharePoint-hosted

Learn more about this choice

New app for SharePoint

Specify the web project type

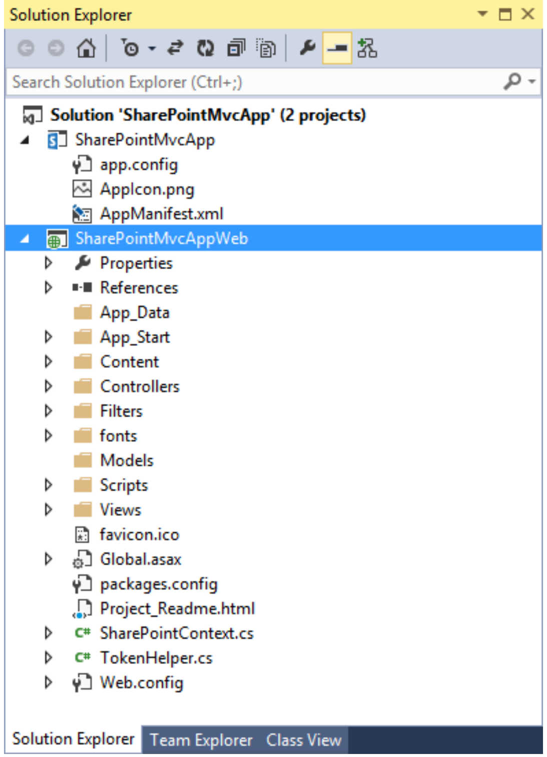
A cloud app for SharePoint consists of an app for SharePoint that is deployed directly to a SharePoint site and a separately deployed web application.

Which type of web application project do you want to create?

○ ASP.NET Web Forms Application

● ASP.NET MVC Web Application

< Previous · Next > · Finish · Cancel



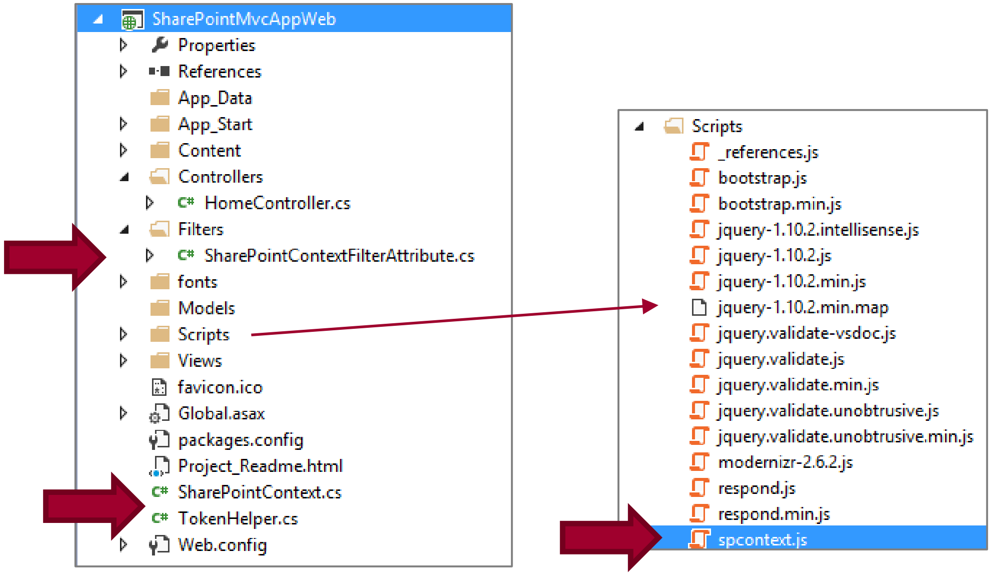
Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'SharePointMvcApp' (2 projects)
  - SharePointMvcApp
    - app.config
    - AppIcon.png
    - AppManifest.xml
  - SharePointMvcAppWeb
    - Properties
    - References
    - App_Data
    - App_Start
    - Content
    - Controllers
    - Filters
    - fonts
    - Models
    - Scripts
    - Views
    - favicon.ico
    - Global.asax
    - packages.config
    - Project_Readme.html
    - SharePointContext.cs
    - TokenHelper.cs
    - Web.config

SharePoint-specific Files

- TokenHelper.cs and SharePointContext.cs
  - Same as discussed earlier in course
- SharePointContextFilterAttribute.cs
  - Provides OAuth redirect logic for when context token is missing
  - Filter has no purpose when using app authentication based on S2S trust
- spcontext.js
  - Client-side JavaScript code to propagate SPHostUrl query string parameter



```
SharePointMvcAppWeb
  Properties
  References
  App_Data
  App_Start
  Content
  Controllers
    HomeController.cs
  Filters
    SharePointContextFilterAttribute.cs
  fonts
  Models
  Scripts
  Views
  favicon.ico
  Global.asax
  packages.config
  Project_Readme.html
  SharePointContext.cs
  TokenHelper.cs
  Web.config
```

```
Scripts
  _references.js
  bootstrap.js
  bootstrap.min.js
  jquery-1.10.2.intellisense.js
  jquery-1.10.2.js
  jquery-1.10.2.min.js
  jquery-1.10.2.min.map
  jquery.validate-vsdoc.js
  jquery.validate.js
  jquery.validate.min.js
  jquery.validate.unobtrusive.js
  jquery.validate.unobtrusive.min.js
  modernizr-2.6.2.js
  respond.js
  respond.min.js
  spcontext.js
```

DEMO

Creating a Provider-hosted SharePoint App Using MVC5

Summary

- ✓ Getting Started with Provider-hosted Apps
- ✓ User Interface Design for the Remote Web
- ✓ Working with ASP.NET MVC
- ✓ Creating Provider-hosted Apps using MVC5