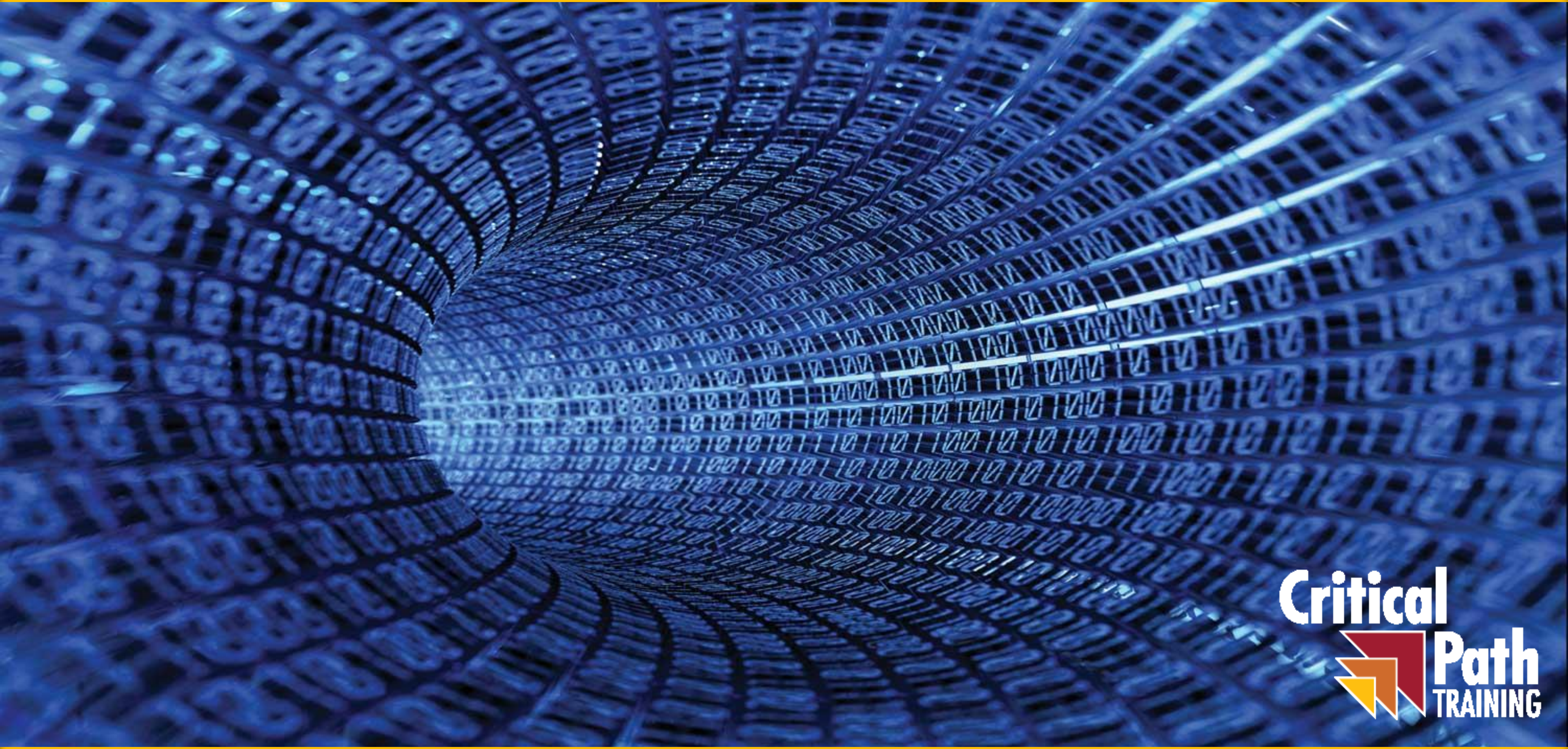
Developing SharePoint Solutions



Critical Path TRAINING

Agenda

- Understanding SharePoint Solutions
- Features and Feature Receivers
- Creating Web Parts
- Creating Site Pages and Application Pages

SharePoint Solutions

- SharePoint development based on solutions
  - Solution is a CAB file with a `*.wsp` extension
  - Solution is a container of files distributed as a unit
  - Solution contain manifest with instructions for installer

- Solutions can be deployed at two different scopes
  - At farm scope as a farm solution *(aka Full Trust Solution)*
  - At site collection scope as a sandboxed solution

Sandbox Solutions

- A short-lived strategy for safer developer extensibility
  - Introduced in SharePoint 2010
  - Solutions with “User Code” are deprecated in SharePoint 2013
  - SharePoint App Model designed to replace Sandbox Solutions
  - Avoid using Sandboxed solutions on any new projects
- No-code sandboxed solutions (NCSS) are not deprecated
  - NCSS created using features and declarative XML elements
  - NCSS can be deployed at either farm level or site collection level
  - NCSS can be used to create lists, site columns and content types
  - NCSS can be used to add content

SharePointRoot Directory

- SharePoint Foundation relies on set of template files
  - Stored in special directory known as SharePointRoot
  - SharePointRoot located on file system of each WFE at this path
    - `C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\16`
  - Farm solutions deploy their files into child directories

Path relative to SharePoint Root	Template file types
/ISAPI	Web Services (*.svc, *.ashx and *.asmx)
/Resources	Resource files (*.resx)
/TEMPLATE/ADMIN	Application pages used exclusively in Central Administration
/TEMPLATE/CONTROLTEMPLATES	ASP.NET User Controls (*.ascx)
/TEMPLATE/FEATURES	Feature definition files (*.xml)
/TEMPLATE/IMAGES	Images (*.gif, *.jpg and *.png)
/TEMPLATE/LAYOUTS	Application pages (*.aspx)
/TEMPLATE/LAYOUTS/1033/STYLES	CSS Files (*.css)
/TEMPLATE/LAYOUTS/ClientBin	Silverlight components (*.xap)
/TEMPLATE/SiteTemplates	Site Definition files (onet.xml)
/TEMPLATE/XML	Custom field type definition files (fldtype*.xml)

Deployment Using Solution Packages

- What is a solution package?
  - Deployment mechanism
  - Atomic unit of reuse, deployment and versioning
  - A set of files and manifest with installation instructions
  - A CAB file with `*.wsp` extension
- What can be deployed via a solution package
  - Feature definitions
  - Images
  - Assemblies
  - And much more…

The manifest.xml file

- Each Solution Package requires `manifest.xml` file
  - Mainly serves as instructions to installer on WFE

```xml
<?xml version="1.0" encoding="utf-8"?>
<Solution xmlns="http://schemas.microsoft.com/sharepoint/"
          SolutionId="07752644-45b2-41c3-9eaa-2d58a1ac31b9"
          SharePointProductVersion="16.0"
          DeploymentServerType="WebFrontEnd"
          ResetWebServer="TRUE">

  <FeatureManifests>
    <FeatureManifest Location="LeadTracker\Feature.xml" />
  </FeatureManifests>

  <TemplateFiles>
    <TemplateFile Location="IMAGES\WingtipDevProject\FeatureIcon.gif" />
    <TemplateFile Location="IMAGES\WingtipDevProject\SiteIcon.gif" />
  </TemplateFiles>

  <Assemblies>
    <Assembly Location="WingtipDevProject.dll" DeploymentTarget="GlobalAssemblyCache" />
  </Assemblies>

</Solution>
```

Farm Solution Deployment

- Done using Windows PowerShell scripts
  - **`Add-SPSolution`** – uploads solution package
  - **`Install-SPSolution`** – deploy solution package

```
Add-PSSnapin Microsoft.SharePoint.Powershell -ErrorAction SilentlyContinue

$SolutionPackageName  = "WingtipDevProject1.wsp"
$SolutionPackagePath = "WingtipDevProject1_v1000\WingtipDevProject1.wsp"

$solution = Get-SPSolution | where-object {$_.Name -eq $SolutionPackageName}
if ($solution -ne $null) {
  if($solution.Deployed -eq $true){
    Uninstall-SPSolution -Identity $SolutionPackageName -Local -Confirm:$false
  }
  Remove-SPSolution -Identity $SolutionPackageName -Confirm:$false
}

Write-Host "Installing Solution..."
Add-SPSolution -LiteralPath $SolutionPackagePath

Install-SPSolution -Identity $SolutionPackageName -Local -GACDeployment
Write-Host "Deployment Complete"
```

Updating a Farm Solution

- Used to push out new files to WFE
  - Used to replace images or DLLs with new version
  - Used in feature upgrade

```
$SolutionPackageName = "WingtipDevProject1.wsp"
$SolutionPackagePath = "WingtipDevProject1_v2000\WingtipDevProject1.wsp"

Update-SPSolution -Identity $SolutionPackageName -LiteralPath
$SolutionPackagePath -Local -GACDeployment
```

- Watch out…
  - Solution update doesn't automatically upgrade features

DEMO

Creating Farm Solutions

Agenda

- ✓ Understanding SharePoint Solutions
- ➢ Features and Feature Receivers
- Creating Web Parts
- Creating Site Pages and Application Pages

Designing and Implementing Features

- What is a SharePoint Feature?
  - Formally known as a “feature definition”
  - A unit of design and implementation
  - A building block for creating SharePoint solutions

- Features can contain elements
  - e.g. menu items, links, list types and list instances
  - Many other element types possible

- Features can contain event handlers
  - Implemented using a feature receiver class
  - Event handler code can program using SharePoint OM

The feature.xml file

- `feature.xml` serves as feature manifest file
  - Defines attributes for feature definition
  - Can reference one or more element manifests
  - Can reference a feature receiver

```xml
<Feature xmlns="http://schemas.microsoft.com/sharepoint/"
         Id="86689158-7048-4421-AD21-E0DEF0D67C81"
         Title="Wingtip Lead Tracker"
         Description="A sample feature deployed using WingtipDevProject1.wsp"
         Version="1.0.0.0"
         Scope="Web"
         Hidden="FALSE"
         ReceiverAssembly="WingtipDevProject1, Version=1.0.0.0, Culture=neutral, PublicKeyToken=1234abcd"
         ReceiverClass="WingtipDevProject1.FeatureReceiver"
         ImageUrl="WingtipDevProject1/FeatureIcon.gif">

  <ElementManifests>
    <ElementManifest Location="elements.xml" />
  </ElementManifests>

</Feature>
```

Element Manifest Files

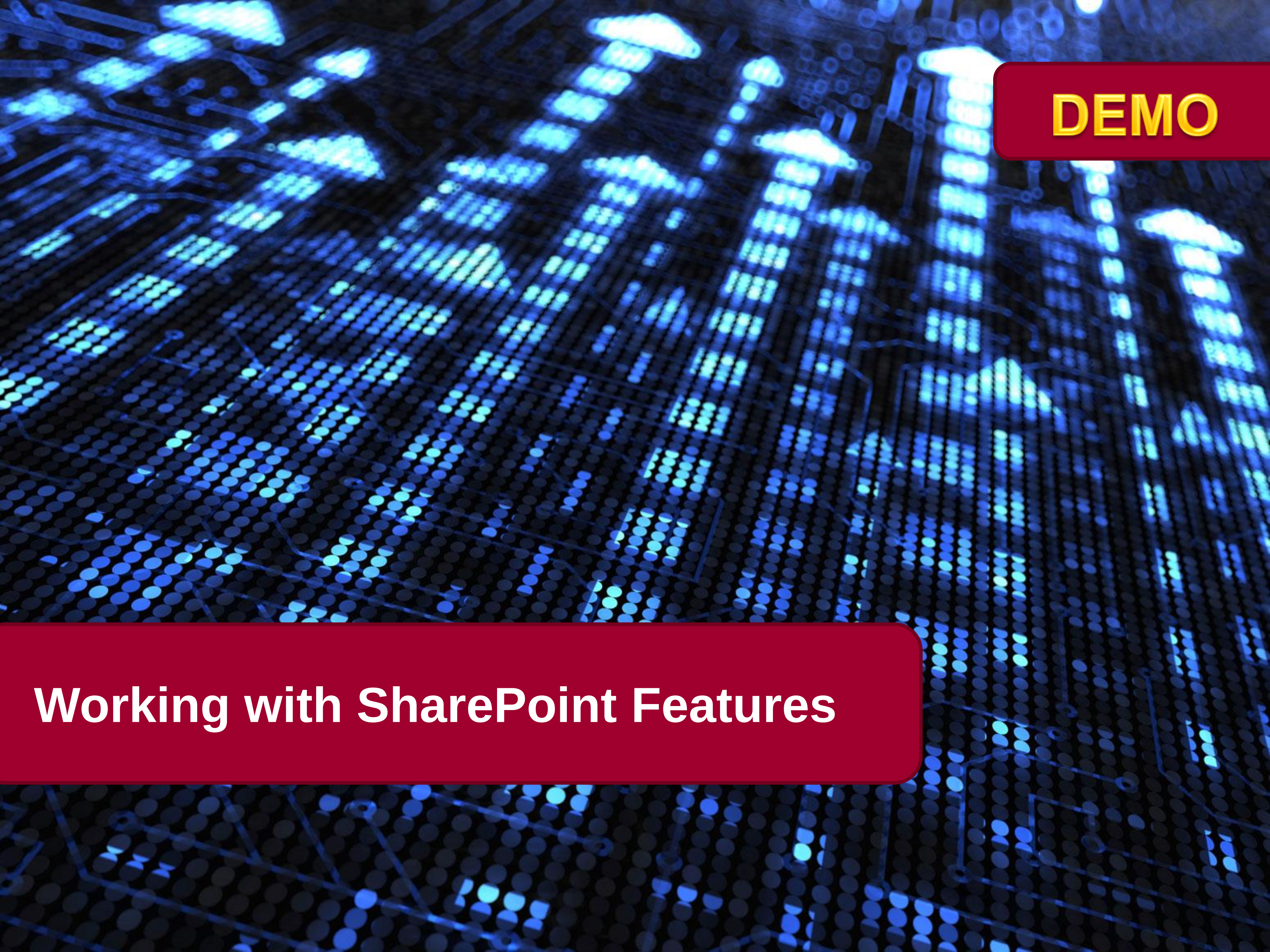
- Element manifest contain declarative elements
  - `ListInstance` elements creates list during activation
  - Many other element types available
  - Element manifest can contain many elements
  - `feature.xml` file can reference many element manifests

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <ListInstance Id="SalesLeads"
    FeatureId="00BFEA71-7E6D-4186-9BA8-C047AC750105"
    TemplateType="105"
    Title="Sales Leads"
    Url="SalesLeads"
    OnQuickLaunch="TRUE" />
</Elements>
```

Feature Receivers

- Feature receiver used to add event handlers
- Must derive from **`SPFeatureReceiver`**
- Not available with Features included in Apps

```
public class FeatureReceiver : SPFeatureReceiver {
  public override void FeatureActivated(SPFeatureReceiverProperties props) {
    SPWeb site = props.Feature.Parent as SPWeb;
    if (site != null) {
      site.Title = "Feature Activated";
      site.SiteLogoUrl = @"_layouts/images/WingtipDevProject1/SiteIcon.gif";
      site.Update();
  } }

  public override void FeatureDeactivating(SPFeatureReceiverProperties props) {
    SPWeb site = props.Feature.Parent as SPWeb;
    if (site != null) {
      site.Title = "Feature Deactivated";
      site.SiteLogoUrl = "";
      site.Update();
      SPList list = site.Lists.TryGetList("Sales Leads");
      if (list != null) { list.Delete(); }
} } }
```



DEMO

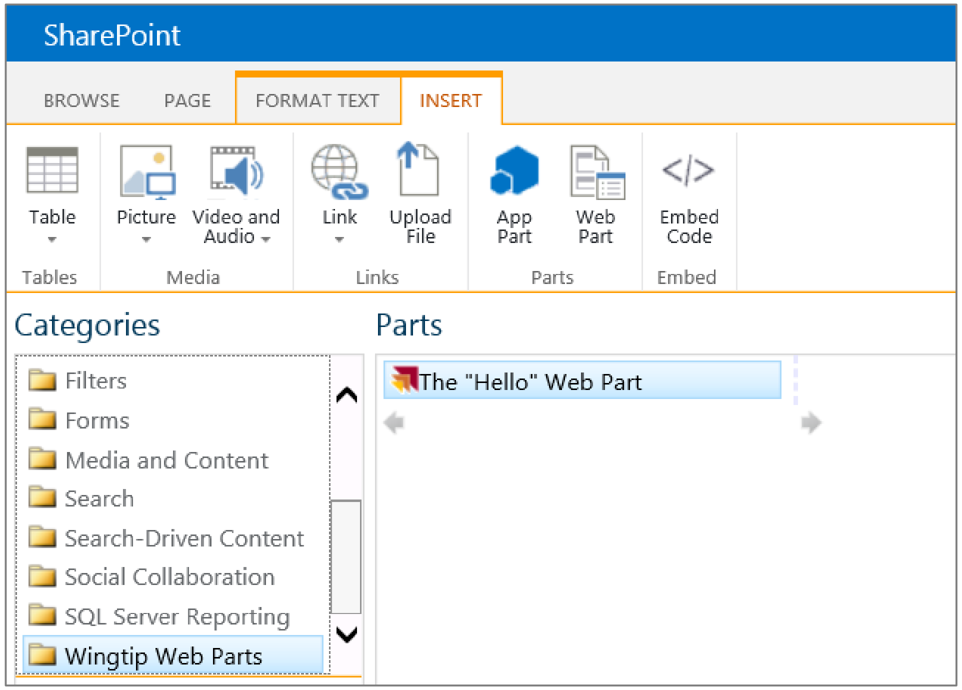
Working with SharePoint Features

Agenda

- ✓ Understanding SharePoint Solutions
- ✓ Features and Feature Receivers
- ➢ Creating Web Parts
- Creating Site Pages and Application Pages

Web Parts

- Web Parts add content and functionality
  - Content is modular, consistent and easy to navigate
  - Configurable chrome: border and title bar
  - Added and configured by users inside browser



Categories

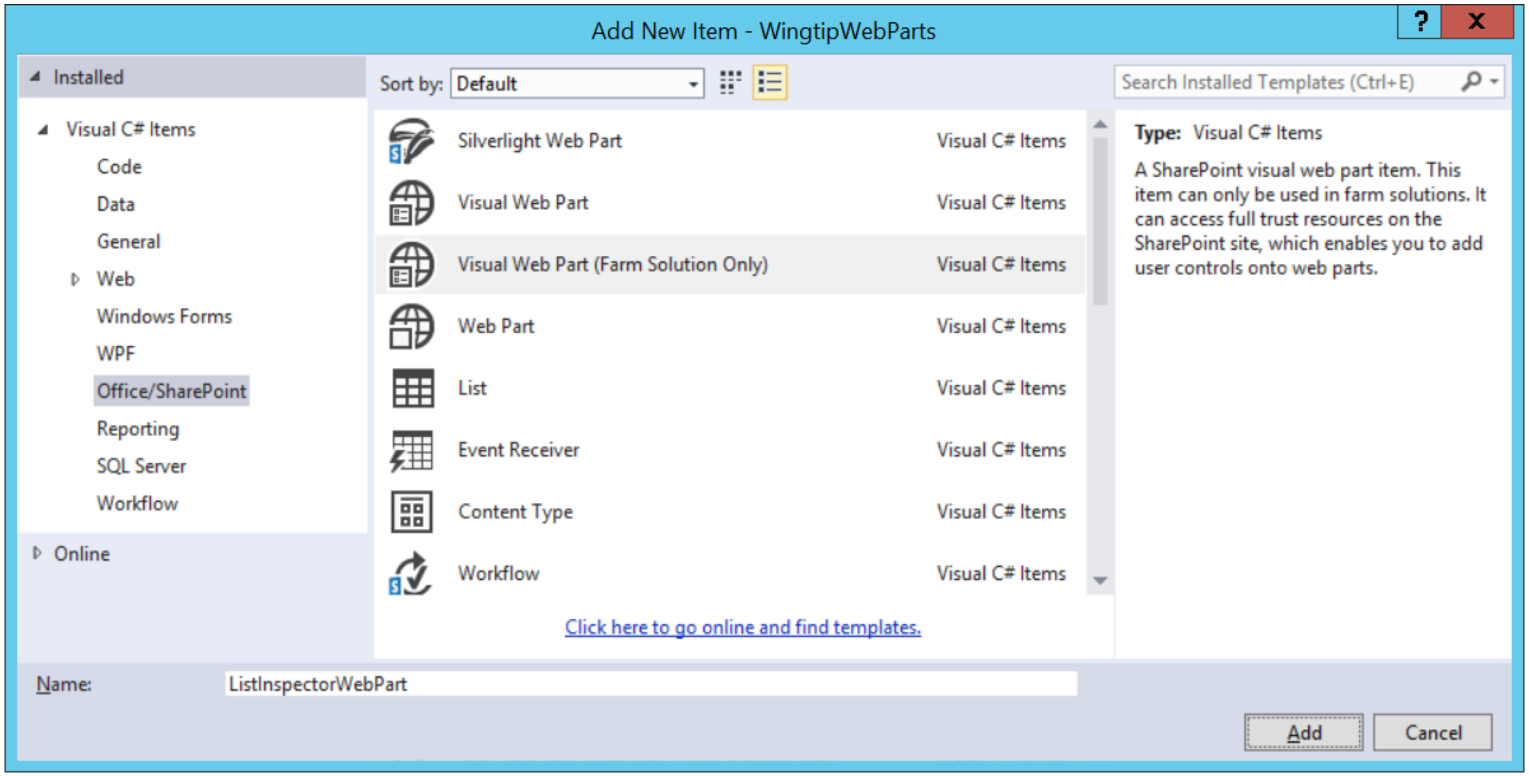
- Filters
- Forms
- Media and Content
- Search
- Search-Driven Content
- Social Collaboration
- SQL Server Reporting
- Wingtip Web Parts

Parts

- The "Hello" Web Part

Overview of Developing Web Parts

- SharePoint Developer Tools supports two types
  - **Web Part:** ASP.NET server control style
  - **Visual Web Part:** ASP.NET user control style (*.ASCX)



Add New Item - WingtipWebParts

Installed
- Visual C# Items
  - Code
  - Data
  - General
  - Web
  - Windows Forms
  - WPF
  - Office/SharePoint
  - Reporting
  - SQL Server
  - Workflow
- Online

Sort by: Default

Search Installed Templates (Ctrl+E)

Silverlight Web Part	Visual C# Items
Visual Web Part	Visual C# Items
Visual Web Part (Farm Solution Only)	Visual C# Items
Web Part	Visual C# Items
List	Visual C# Items
Event Receiver	Visual C# Items
Content Type	Visual C# Items
Workflow	Visual C# Items

Click here to go online and find templates.

Type: Visual C# Items

A SharePoint visual web part item. This item can only be used in farm solutions. It can access full trust resources on the SharePoint site, which enables you to add user controls onto web parts.

Name: ListInspectorWebPart

DEMO

Creating a Custom Web Part

Agenda

- ✓ Understanding SharePoint Solutions
- ✓ Features and Feature Receivers
- ✓ Creating Web Parts
- ➢ Creating Site Pages and Application Pages

SharePoint Sites are Collections of Pages

- All pages share the same look and feel
  - HTML page layout defined by common Master Page
  - Page formatting defined by common CSS files
  - Client-side behaviors defined by JavaScript files
- Pages within site can be split into two categories
  - **Site Pages** exist within the content DB for a site
  - **Application Pages** only exist on file system of WFE

Site Pages vs. Application Pages

- Site Pages exist within virtual file system of site
  - They support customization via Web Parts and/or SPD
  - Site pages can be rendered using underlying template
  - Page using template is said to be a ghosted page
  - Page can be customized (unghosted) by user
- Application Pages are deployed once per farm
  - They are accessible through _**layouts** virtual directory
  - They are parsed / compiled in classic ASP.NET mode
  - They do not support any form of user customization
  - The can only be added using farm solutions

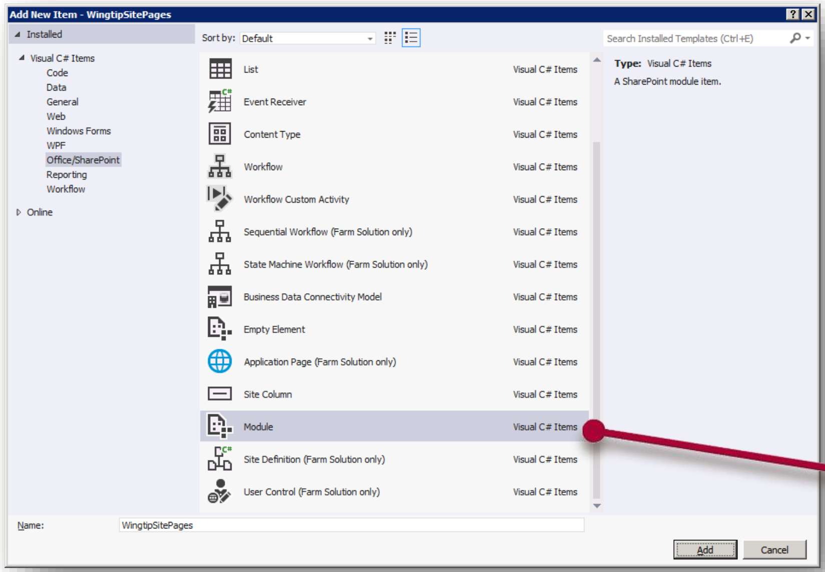
Site Pages Overview

- Site pages can be added by a developer
  - Works in SharePoint solutions and in SharePoint apps
  - Sites pages and related resources added with **Modules**
  - Site pages cannot contain any server-side code

- Developers create site pages using templates
  - You create a Module and add one of more templates
  - Feature activation create instances from the templates
  - Sites pages initially in uncustomized state (ghosted)
  - Sites pages can be customized (unghosted) by users

Creating Site Pages from Page Templates

- Site pages created using a **Module**
  - Module must be associated with a feature
  - Visual Studio adds project folder with `elements.xml` file
  - Inside is a `<Module>` element with `<File>` elements



Add New Item - WingtipSitePages

Installed
- Visual C# Items
  - Code
  - Data
  - General
  - Web
  - Windows Forms
  - WPF
  - Office/SharePoint
  - Reporting
  - Workflow
- Online

Sort by: Default

Search Installed Templates (Ctrl+E)

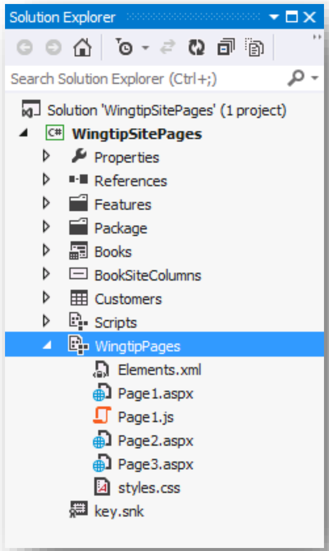
List	Visual C# Items
Event Receiver	Visual C# Items
Content Type	Visual C# Items
Workflow	Visual C# Items
Workflow Custom Activity	Visual C# Items
Sequential Workflow (Farm Solution only)	Visual C# Items
State Machine Workflow (Farm Solution only)	Visual C# Items
Business Data Connectivity Model	Visual C# Items
Empty Element	Visual C# Items
Application Page (Farm Solution only)	Visual C# Items
Site Column	Visual C# Items
Module	Visual C# Items
Site Definition (Farm Solution only)	Visual C# Items
User Control (Farm Solution only)	Visual C# Items

Type: Visual C# Items

A SharePoint module item.

Name: WingtipSitePages

Add   Cancel



Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'WingtipSitePages' (1 project)
  - **WingtipSitePages**
    - Properties
    - References
    - Features
    - Package
    - Books
    - BookSiteColumns
    - Customers
    - Scripts
    - WingtipPages
      - Elements.xml
      - Page1.aspx
      - Page1.js
      - Page2.aspx
      - Page3.aspx
      - styles.css
    - key.snk

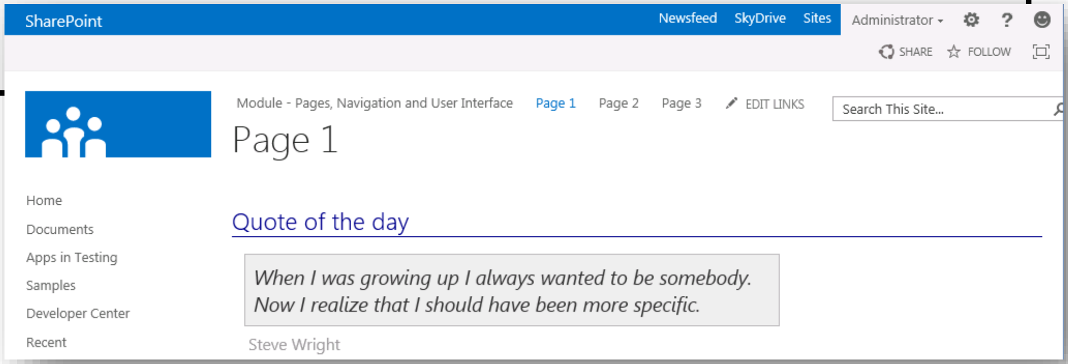
Modules & Elements.xml File

- Visual Studio updates Module `element.xml` for you
  - You just create / add files to Module folder
  - Some scenarios requires manual edits to `elements.xml`

```xml
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <Module Name="WingtipPages">
    <File Path="WingtipPages\Page1.aspx" Url="WingtipPages/Page1.aspx" />
    <File Path="WingtipPages\Page2.aspx" Url="WingtipPages/Page2.aspx" />
    <File Path="WingtipPages\Page3.aspx" Url="WingtipPages/Page3.aspx" />
    <File Path="WingtipPages\styles.css" Url="WingtipPages/styles.css" />
  </Module>
</Elements>
```

'Hello World' Page Template for Site Page

```
<%@ Page MasterPageFile="~masterurl/default.master" %>

<asp:Content ContentPlaceHolderId="PlaceHolderAdditionalPageHead" runat="server">
  <link href="styles.css" rel="stylesheet" type="text/css" />
  <script src="Page1.js" type="text/javascript"></script>
</asp:Content>

<asp:Content ContentPlaceHolderId="PlaceHolderPageTitle" runat="server">
  Page 1 – This shows up at the top of the browser window
</asp:Content>

<asp:Content ContentPlaceHolderId="PlaceHolderPageTitleInTitleArea" runat="server">
  Page 1
</asp:Content>

<asp:Content ContentPlaceHolderId="PlaceHolderMain" runat="server">
  <h3>Quote of the day</h3>
  <p id="quote">When I was growing up I always wanted to be somebody.
                Now I realize that I should have been more specific.</p>
  <p id="quote_author">Steve Wright</p>
</asp:Content>
```



Page 1

Quote of the day

When I was growing up I always wanted to be somebody.
Now I realize that I should have been more specific.

Steve Wright

Designing Web Part Pages

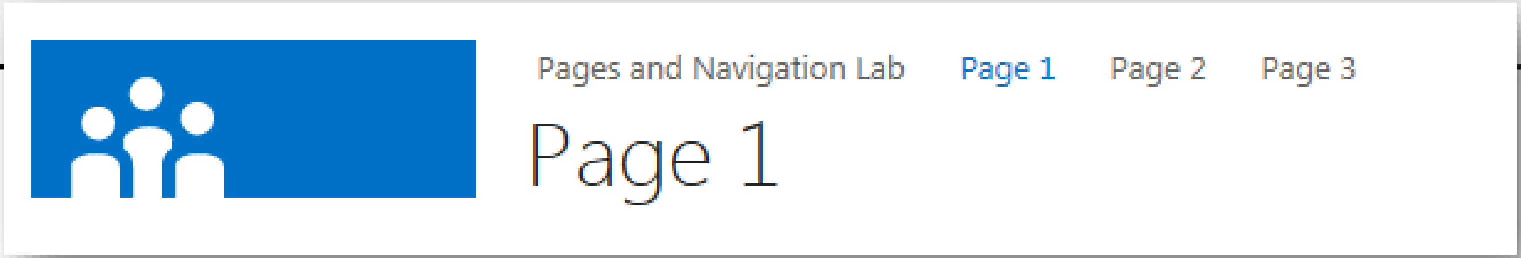
- Changes from previous page templates
  - Inherit from `WebPartPage` class
  - Add controls for web part zones and web parts

```
<%@ Assembly Name="Microsoft.SharePoint, Version=15.0.0.0, Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>

<%@ Page MasterPageFile="~masterurl/default.master" Inherits="Microsoft.SharePoint.WebPartPages.WebPartPage" %>

<%@ Register TagPrefix="WebPartPages"
  Namespace="Microsoft.SharePoint.WebPartPages"
  Assembly="Microsoft.SharePoint, Version=15.0.0.0, Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>

<asp:Content ContentPlaceHolderID="PlaceHolderMain" runat="server">
  <WebPartPages:WebPartZone ID="Main" Title="Main Web Part Zone" FrameType="TitleBarOnly" runat="server">
    <ZoneTemplate>
      <WebPartPages:XsltListViewWebPart
              runat="server" ID="CsutomersWebPart"
              Title="Customers" ListUrl="Lists/Customers"
              ChromeType="None">
      </WebPartPages:XsltListViewWebPart>
    </ZoneTemplate>
  </WebPartPages:WebPartZone>
</asp:Content>
```

Adding Navigation Nodes to Top Nav Bar

- Simple navigation technique for teams sites
  - Done using server-side code or client-side code
  - Not a technique to use in publishing sites

```
public class MainSiteEventReceiver : SPFeatureReceiver
{
  public override void FeatureActivated(SPFeatureReceiverProperties properties)
  {
    SPSite siteCollection = properties.Feature.Parent as SPSite;
    if (siteCollection != null)
    {
      SPWeb site = siteCollection.RootWeb;
      // create menu items on top link bar for custom site pages
      SPNavigationNodeCollection topNav = site.Navigation.TopNavigationBar;
      topNav.AddAsLast(new SPNavigationNode("Page 1", "WingtipPages/Page1.aspx"));
      topNav.AddAsLast(new SPNavigationNode("Page 2", "WingtipPages/Page2.aspx"));
      topNav.AddAsLast(new SPNavigationNode("Page 3", "WingtipPages/Page3.aspx"));
    }
  }
}
```

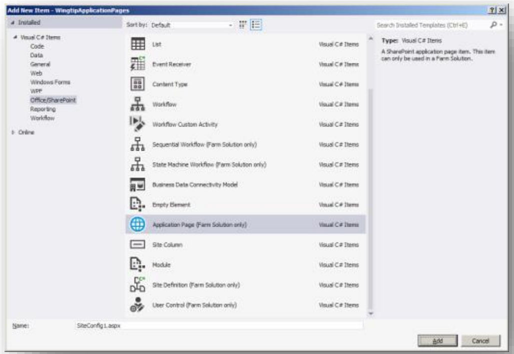


DEMO

Provisioning Site Pages using Page Templates

Creating Application Pages

- Creating custom application pages
  - Visual Studio provides project item template
  - Only supported in farm solutions
  - Only type of page that supports server-side code


Add New Item - WingtipApplicationPages

Installed
- Visual C# Items
  - Code
  - Data
  - General
  - Web
  - Windows Forms
  - WPF
  - Office/SharePoint
  - Reporting
  - Workflow
- Online

Sort by: Default

Search Installed Templates (Ctrl+E)

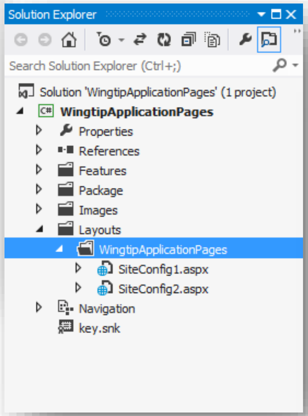
List	Visual C# Items
Event Receiver	Visual C# Items
Content Type	Visual C# Items
Workflow	Visual C# Items
Workflow Custom Activity	Visual C# Items
Sequential Workflow (Farm Solution only)	Visual C# Items
State Machine Workflow (Farm Solution only)	Visual C# Items
Business Data Connectivity Model	Visual C# Items
Empty Element	Visual C# Items
Application Page (Farm Solution only)	Visual C# Items
Site Column	Visual C# Items
Module	Visual C# Items
Site Definition (Farm Solution only)	Visual C# Items
User Control (Farm Solution only)	Visual C# Items

Type: Visual C# Items

A SharePoint application page item. This item can only be used in a Farm Solution.

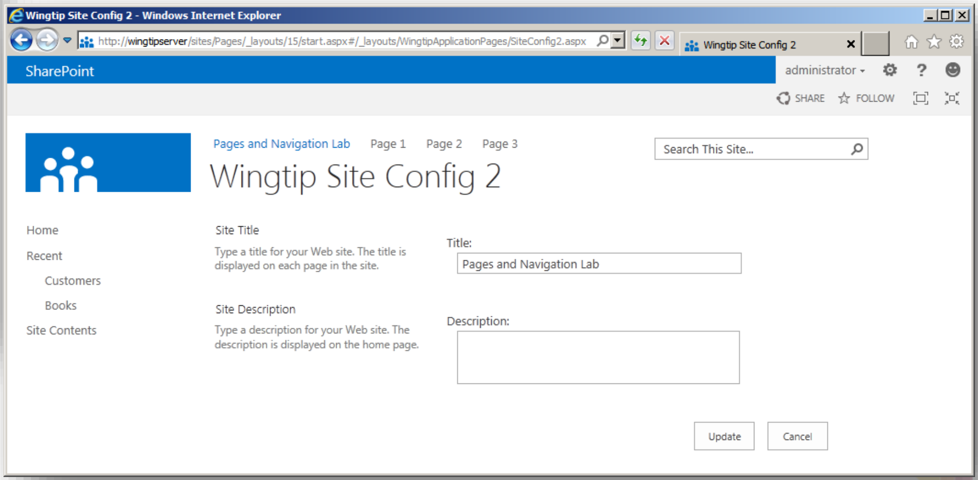
Name: SiteConfig1.aspx

Add Cancel


Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'WingtipApplicationPages' (1 project)
  WingtipApplicationPages
    Properties
    References
    Features
    Package
    Images
    Layouts
      WingtipApplicationPages
        SiteConfig1.aspx
        SiteConfig2.aspx
    Navigation
    key.snk
```


Wingtip Site Config 2 - Windows Internet Explorer

http://wingtipserver/sites/Pages/_layouts/15/start.aspx#/_layouts/WingtipApplicationPages/SiteConfig2.aspx

Wingtip Site Config 2

SharePoint administrator ▾

SHARE FOLLOW

Pages and Navigation Lab Page 1 Page 2 Page 3

Search This Site...

Wingtip Site Config 2

Home
Recent
Customers
Books
Site Contents

Site Title

Type a title for your Web site. The title is displayed on each page in the site.

Title: Pages and Navigation Lab

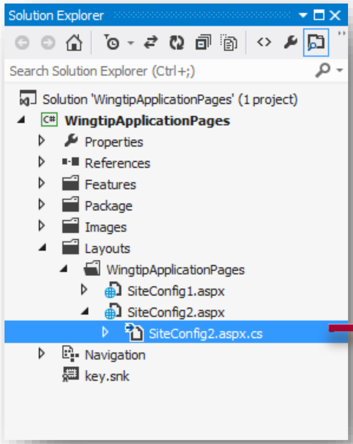
Site Description

Type a description for your Web site. The description is displayed on the home page.

Description:

Update Cancel

Adding Code to An Application Page


Solution Explorer

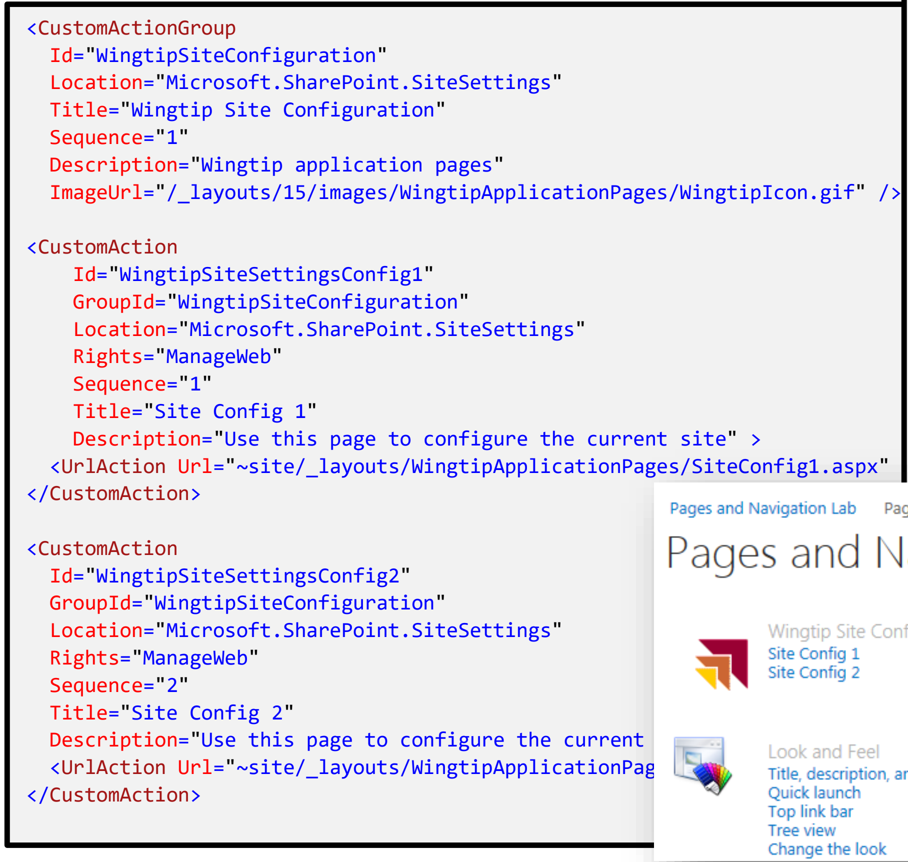
Search Solution Explorer (Ctrl+;)

```
Solution 'WingtipApplicationPages' (1 project)
  WingtipApplicationPages
    Properties
    References
    Features
    Package
    Images
    Layouts
      WingtipApplicationPages
        SiteConfig1.aspx
        SiteConfig2.aspx
          SiteConfig2.aspx.cs
    Navigation
    key.snk
```

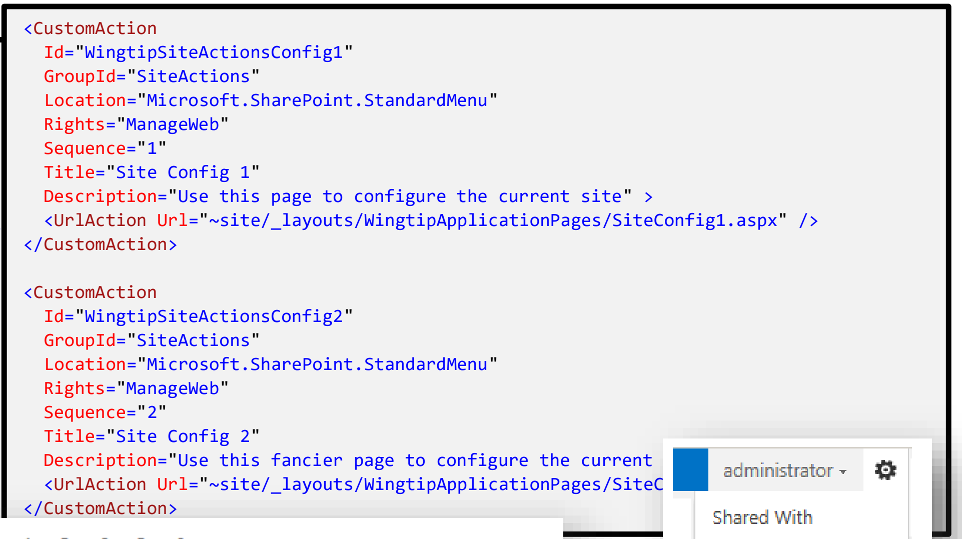
```
public partial class SiteConfig2 : LayoutsPageBase {

  protected override void OnInit(EventArgs e) {
    base.OnInit(e);
    cmdUpdate.Click += new EventHandler(cmdUpdate_Click);
  }

  void cmdUpdate_Click(object sender, EventArgs e) {
    SPWeb site = this.Web;
    site.Title = txtSiteTitle.Text;
    site.Description = txtSiteDescription.Text;
    site.Update();
    SPUtility.Redirect("settings.aspx",
                       SPRedirectFlags.RelativeToLayoutsPage,
                       this.Context);
  }

  protected override void OnPreRender(EventArgs e) {
    base.OnPreRender(e);
    SPWeb site = this.Web;
    txtSiteTitle.Text = site.Title;
    txtSiteDescription.Text = site.Description;
  }
}
```

Navigating with CustomActions

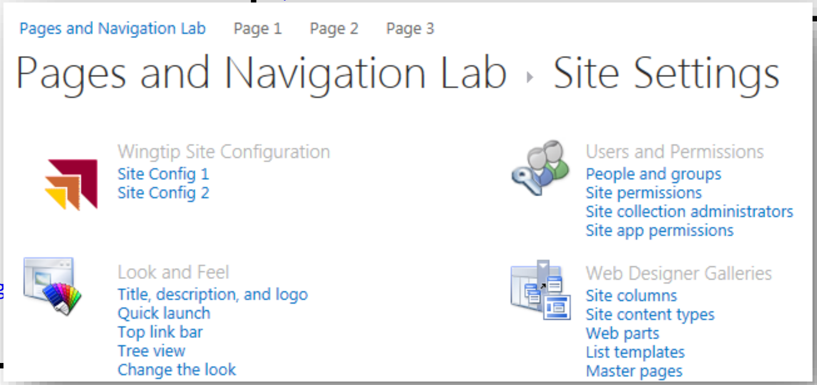
- `CustomAction` elements provide navigation
  - Add Site Settings links and Site Actions menu items



```
<CustomActionGroup
  Id="WingtipSiteConfiguration"
  Location="Microsoft.SharePoint.SiteSettings"
  Title="Wingtip Site Configuration"
  Sequence="1"
  Description="Wingtip application pages"
  ImageUrl="/_layouts/15/images/WingtipApplicationPages/WingtipIcon.gif" />

<CustomAction
    Id="WingtipSiteSettingsConfig1"
    GroupId="WingtipSiteConfiguration"
    Location="Microsoft.SharePoint.SiteSettings"
    Rights="ManageWeb"
    Sequence="1"
    Title="Site Config 1"
    Description="Use this page to configure the current site" >
  <UrlAction Url="~site/_layouts/WingtipApplicationPages/SiteConfig1.aspx"
</CustomAction>

<CustomAction
  Id="WingtipSiteSettingsConfig2"
  GroupId="WingtipSiteConfiguration"
  Location="Microsoft.SharePoint.SiteSettings"
  Rights="ManageWeb"
  Sequence="2"
  Title="Site Config 2"
  Description="Use this page to configure the current
  <UrlAction Url="~site/_layouts/WingtipApplicationPag
</CustomAction>
```



```
<CustomAction
  Id="WingtipSiteActionsConfig1"
  GroupId="SiteActions"
  Location="Microsoft.SharePoint.StandardMenu"
  Rights="ManageWeb"
  Sequence="1"
  Title="Site Config 1"
  Description="Use this page to configure the current site" >
  <UrlAction Url="~site/_layouts/WingtipApplicationPages/SiteConfig1.aspx" />
</CustomAction>

<CustomAction
  Id="WingtipSiteActionsConfig2"
  GroupId="SiteActions"
  Location="Microsoft.SharePoint.StandardMenu"
  Rights="ManageWeb"
  Sequence="2"
  Title="Site Config 2"
  Description="Use this fancier page to configure the current
  <UrlAction Url="~site/_layouts/WingtipApplicationPages/SiteC
</CustomAction>
```



Pages and Navigation Lab   Page 1   Page 2   Page 3

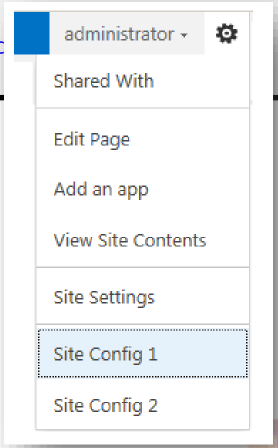
Pages and Navigation Lab › Site Settings

Wingtip Site Configuration
Site Config 1
Site Config 2

Users and Permissions
People and groups
Site permissions
Site collection administrators
Site app permissions

Look and Feel
Title, description, and logo
Quick launch
Top link bar
Tree view
Change the look

Web Designer Galleries
Site columns
Site content types
Web parts
List templates
Master pages



Summary

- ✓ Understanding SharePoint Solutions
- ✓ Features and Feature Receivers
- ✓ Creating Web Parts
- ✓ Creating Site Pages and Application Pages