Microsoft® Official Course

Module 03

Custom Components & Declarations

Microsoft®

Module Overview

- Component Overview
- Defining Custom Lists
- Defining Custom Sites
- Managing SharePoint Sites

Lesson 1: Component Overview

- The Component Hierarchy
- Site Columns
- Site Content Types
- List Columns and Content Types

The Component Hierarchy

- Farm
- Services
- Web application
- Site Collection
- Site
  - Site columns
  - Site content types
- List/Library
  - List columns
  - List content types
- List item
- Field

Site Columns

- Definition not implementation
- Scopes
  - Site Collection
  - Site
- Properties
  - Display name
  - Internal name
  - Data type
  - Group
  - Required
- Create declaratively, by using the object model, or by using the SharePoint UI

Site Columns

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
    <Field ID="{770E459A-CFE4-40BD-873A-945DBD430675}"
        Name="ContosoSiteColumn"
        DisplayName="Contoso Site Column"
        Type="Text"
        Required="FALSE"
        Group="Contoso Columns">
    </Field>
</Elements>
```

```
using(SPWeb web = site.RootWeb) // Assume site is a valid SPSite reference.
{
    string displayName = "Contoso Site Column";
    string internalName = web.Fields.Add(displayName, SPFieldType.Text, false);
    SPField field = web.Fields[displayName];
    field.Group = "Contoso Columns";
    field.Update();
    web.Update();
}
```

Site Content Types

- Collection of columns
- Definition not implementation
- Scopes
  - Site collection
  - Site
- Hierarchical
  - Item
    - Announcement
    - Document
    - Folder
    - Custom
- Create declaratively, by using the object model, or by using the SharePoint UI

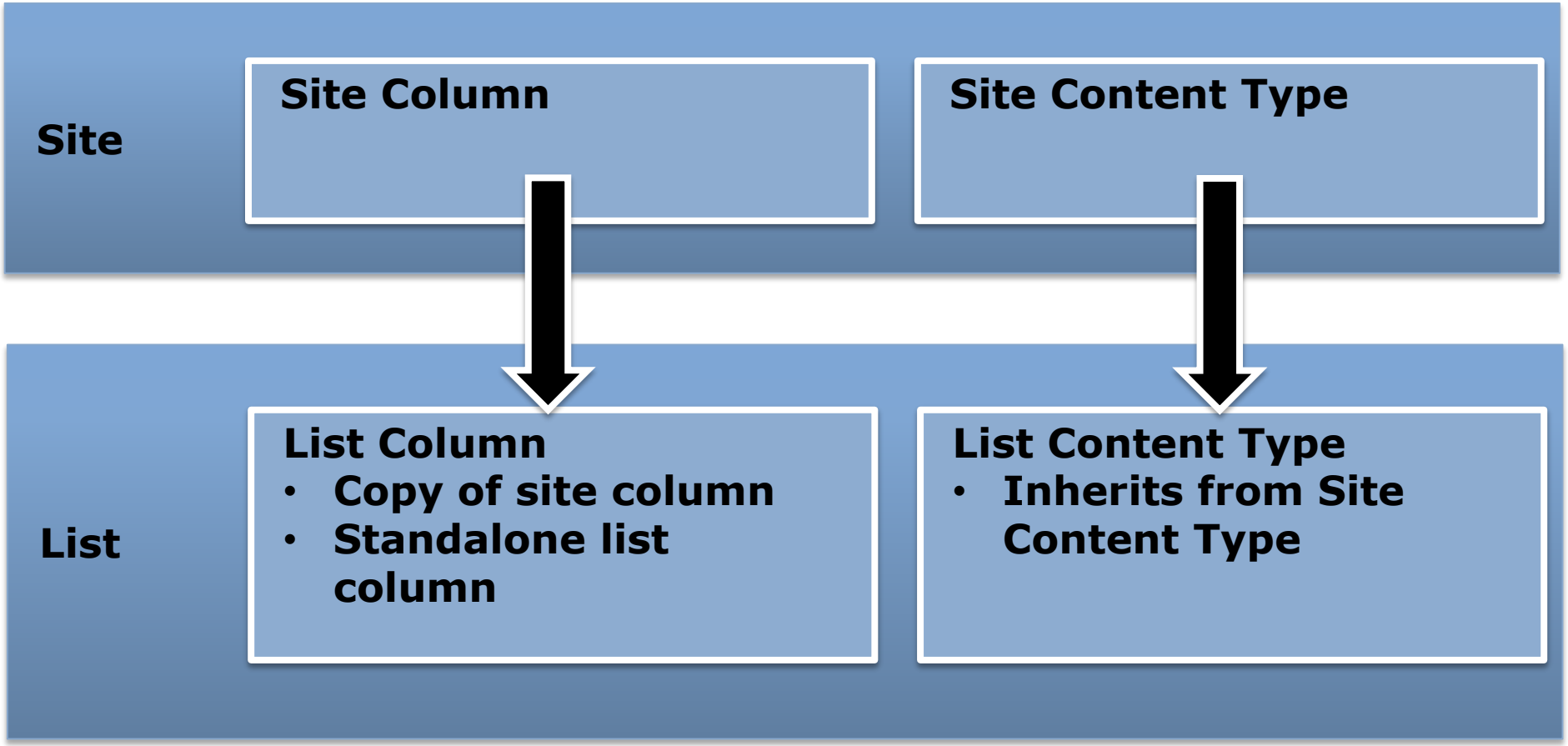
Content Types

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint">
<ContentType ID="0x0100770E459ACFE440BD873A945DBD430675" Name="Assistant"
        Group="Contoso Content Types" Description="A content type to represent an
assistant."
        Inhertis="TRUE", Version="0">
        <FieldRefs>
            <!-- Add existing site columns by reference. You must provide the ID of each
site column. -->
<FieldRef ID="{2AEA194D-E399-4f05-95AF-94F87B1F2687}"
                DisplayName="$Resources:core,Assistants_Name_OL;" Name="AssistantsName"
Sealed="TRUE" />
<FieldRef ID="{FD630629-C165-4513-B43C-FDB16B86A14D}"
                DisplayName="$Resources:core,Business_Phone;" Name="WorkPhone" />
        </FieldRefs>
    </ContentType>
</Elements>
```

Content Types

```
using(SPWeb web = site.RootWeb)
{
    // Obtain a reference to the parent content type.
    SPContentType itemContentType = web.ContentTypes["Item"];
// Create a new content type, by specifying the parent content type, the group to which
the content
    // type will ultimately be added, and name for the new content type.
    SPContentType contentType = new SPContentType(itemContentType, web.ContentTypes,
"Assistant");
    // Set properties on the content type.
    contentType.Description = "A content type to represent an assistant.";
    contentType.Group = "Contoso Content Types";
    // Obtain references to the fields which should be added to the content type.
    SPField nameField = web.AvailableFields["AssistantsName"];
    SPField phoneNumberField = web.AvailableFields["WorkPhone"];
    // Create field links for each of the fields.
    SPFieldLink nameFieldLink = new SPFieldLink(nameField);
    SPFIeldLink phoneNumberFieldLink = new SPFieldLink(phoneNumberField);
    // Add each of the field links to the content type.
    contentType.FieldLinks.Add(nameFieldLink);
    contentType.FieldLinks.Add(phoneNumberFieldLink);
    // Add the new content type to the sites content types collection.
    web.ContentTypes.Add(contentType);
    // Update the content type.
    contentType.Update()
    // Update the site.
    web.Update();
}
```

List Columns and Content Types



Lesson 2: Defining Custom Lists

- Introduction to List Templates
- Developing List Templates
- The Visual Studio List Designer
- Discussion Question
- Provisioning Lists

Introduction to List Templates

- Define list template
  - Content types
  - Fields
  - Views
  - Forms
- Create list instance
  - Declarative (Feature)
  - Programmatic (SharePoint object model)

Developing List Templates

- List template Feature
  - Feature.xml file (in the root of the Feature folder)
  - *Manifest.xml* file (optionally in a sub-folder, can have any name, but must be referenced in Feature.xml file)
  - *ListName* folder (determined by the value of the **Name** attribute of the **ListTemplate** element in the manifest)
    - Schema.xml file

- *Manifest.xml*

```
<Elements>
    <ListTemplate>
```

- *Schema.xml*

```
<List>
    <MetaData>
        <ContentTypes>
        <Fields>
        <Views>
        <Forms>
```

List Definitions

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
    <ListTemplate
        Name="AssistantList {Name for the list template; the name of the sub-folder for the
schema.xml file}"
        Type="10001 {Unique identifier for the list template, use values above 10000 to avoid clashes}"
        BaseType="0 {Identifier of the list on which this list is based, 0 for generic list}"
        OnQuickLaunch="TRUE {Specifies whether list instances should appear on the quick launch by
default}"
        DisplayName="Assistants List {Display name for the list template, displayed when users
create new lists}"
        Description="List to store details of assistants {Description for the list template}"
</Elements>
```

```
<List xmlns="http://schemas.microsoft.com/sharepoint" … >
    <MetaData>
        <ContentTypes>
            <ContentType ID="…" Name="…"> … </ContentType>
            <ContentTypeRef ID="…" … />
        </ContentTypes>
        <Fields>
            <Field ID="…" Name="…" … />
            <FieldRef ID="…" Name="…" />
        </Fields>
        <Views>
            <View … > … </View>
        </Views>
        <Forms>
            <Form Type="DisplayForm" Url="DispForm.aspx" … />
        </Forms>
    </Metadata>
</List>
```

The Visual Studio List Designer

- Manage
  - Columns
  - Content types
  - Views
  - Title
  - Url
  - Description
- Faster (select from lists, no need to lookup GUIDs)
- Easier (no need to write XML)
- Reduced risk (less chance of typographical error)

Provisioning Lists

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
    <ListInstance Title="Products" TemplateType="10001"
        Description="A list to store product information."
        Url="Lists/Products">
    </ListInstance>
</Elements>
```

```
using(SPWeb web = site.RootWeb)
{
// Obtain a reference to the list template by using the ListTemplates collection, and by
specifying the
    // template name.
    SPListTemplate template = web.ListsTemplates["ProductListTemplate"];
    // Create a new instance of the list template.
    Guid productsList = web.Lists.Add("Products", "A list to store product information.",
template);
//Create a new instance of the Announcements list template by using the
SPListTemplateType
    // enumeration to identify the list template.
Guid announcementsList = web.Lists.Add("Announcements",  "A list to store
announcements.",
        SPListTemplateType.Announcements);
    // Call the Update method on the SPWeb instance.
    web.Update();
}
```

Lesson 3: Defining Custom Sites

- Introduction to Site Definitions and Web Templates
- Developing a Site Definition Declaratively
- Deploying Site Definitions
- Using Code to Provision a Custom Site
- Demonstration: Examining the Standard Site Definitions
- Feature Stapling
- Discussion Question

Introduction to Site Definitions and Web Templates

- Site definition
  - Deploy with a farm solution
  - Not dependent on any other component
  - Time consuming to develop
  - Cached by the IIS worker process for performance
- Web template
  - Deploy with a sandboxed or farm solution
  - Dependent on source site definition
  - Fast to develop – SPWeb.SaveAsTemplate
  - Not cached for performance

Developing a Site Definition Declaratively

- Webtemp*.xml
  - 15\TEMPLATES\SiteTemplates
  - Template and configuration definitions
- Onet.xml
  - Site content and structure
- Supporting files (pages, images, and so on)
- Assemblies

Site Template

```
<Project xmlns="http://schemas.microsoft.com/sharepoint/"  Title="ContosoSiteDefinition">
    <NavBars>
        <!-- Add NavBar and NavBarLink elements here to add navigation bars and links to
your site. -->
    </NavBars>
    <Configurations>
        <!-- Add a Configuration element for each configuration defined in the
webtemp*.xml file here. -->
        <Configuration ID="0" Title="SalesSiteDefintion">
            <Lists>
                <!-- Add lists to include in this configuration here. -->
            </Lists>
            <SiteFeatures>
                <!-- Add site collection Features to include in this configuration here.
-->
            </SiteFeatures>
            <WebFeatures>
                <!-- Add site Features to include in this configuration here. -->
            </WebFeatures>
            <Modules>
                <!-- Add modules to include in this configuration here. -->
            </Modules>
        </Configuration>
    </Configurations>
    <Modules>
        <!-- Add Module elements here, for example to include pages or files in your site
definition. -->
    </Modules>
    <Components>
        <!-- Add custom components such as an external security provider here. -->
    </Components>
    <ServerEmailFooter>
        <!-- Specify the server email footer here. -->
    </ServerEmailFooter>
</Project>
```

Using Code to Provision a Custom Site

```
public class ContosoSite : SPWebProvisioningProvider
{
    public override Provision(SPWebProvisioningProperties props)
    {
        // Obtain a reference to the new SPWeb object.
        SPWeb newWeb = props.Web;
        // Apply the blank site template.
        newWeb.ApplyWbTemplate("STS#1");
        // Retrieve configuration data.
        string data = props.Data;
        // Add lists, libraries, Features, pages, and other components to the site.
        // web.Lists.Add(…);
        // web.Features.Add(…);
    }
}
```

```
<Templates xmlns:ows="Microsoft.SharePoint">
    <Template Name="ContosoSite" ID="12001">
        <Configuration ID="0"
            Title="Contoso Research Site"
            Hidden="FALSE"
            ImageUrl="/_layouts/15/ContosoSiteImage.png"
            Description="This is a custom site template for Contoso research projects."
            DisplayCategory="Contoso Sites"
ProvisionAssembly=
                "ContosoAssembly, Version=1.0.0.0, Culture=neutral,
PublicKeyToken=abcde1234567890a"
            ProvisionClass="Contoso.ContosoSite"
            ProvisionData="ResearchConfiguration.xml"
            RootWebOnly="FALSE"
    <Template>
</Templates>
```

Demonstration: Examining the Standard Site Definitions

- In this demonstration, you will review:
  - The site definitions included in SharePoint 2013
  - The webtemp.xml files
  - The onet.xml files

Feature Stapling

- Add a Feature to an existing site definition
  - Out-of-the-box site definitions
  - Custom site definitions that are in use
- Stapled Feature
- Stapler Feature

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
    <FeatureSiteTemplateAssociation
        Id="3E00ABF6-1301-4698-9BD1-5500725000F0"
        TemplateName="STS#0"
</Elements>
```

Discussion Question

When would you choose to create a site definition, a web template, or use Feature stapling?

Lesson 4: Managing SharePoint Sites

- Understanding the Site Hierarchy
- Provisioning Sites
- Host-named Site Collections
- Backing-up Site Collections
- Managing the Site Lifecycle
- Upgrading SharePoint 2010 Sites

Understanding the Site Hierarchy

- SPFarm.Services
- SPWebService.WebApplications
- SPWebApplication.Sites
- SPSite.AllWebs
- SPWeb

Provisioning Sites

- SharePoint object model
  - SPSite site = webApplication.Sites.Add(…);
  - SPWeb web= site.AllWebs.Add(…);
- PowerShell
  - New-SPSite
  - New-SPWeb
- SharePoint user interface

Host-named Site Collections

- Web Application A (port 80)
  - http://sales.contoso.com (site collection)
    - http://sales.contoso.com/sites/subsite (site)
    - http://sales.contoso.com/sites/subsite2 (site)
  - http://research.contoso.com (site collection)
  - http://sales.contoso.com/sites/sitecollection (site collection)
- Web Application 2 (IIS host header – sharepoint.contoso.com)
  - http://sharepoint.contoso.com (site collection)
  - http://sharepoint.contoso.com/sites/sitecollection (site collection)
  - http://sharepoint.contoso.com/site/subsite (site)

Managing the Site Lifecycle

- Lifecycle events
  - WebAdding
  - WebProvisioned
  - WebMoving
  - WebMoved
  - WebDeleting
  - WebDeleted
  - SiteDeleted
- Identify inactive site collections
  - SPSite.LastContentModifiedDate
- Delete sites and site collections
  - SPWeb.Delete()
  - SPSite.Delete()

Managing Site Collections

```
using(SPSite site = new SPSite("http://sales.contoso.com"))
{
    // Obtain a reference to the SPSiteCollection which contains this SPSite.
    // The SPSiteCollection is exposed as a property of the current SPSite instance's
parent web application.
    SPSitemCollection siteCollection = site.WebApplication.Sites;
    // Backup the site collection.
    siteCollection.Backup(site.Url, @"C:\Backups\SiteBackup.backup", false);
    // Restore the site collection (overwrite the existing site collection).
    siteCollection.Restore(site.Url, @"C:\Backups\SiteBackup.backup", true);
}
```

```
using(SPSite site = new SPSite("http://inactive.contoso.com"))
{
    int daysSinceLastChange = (DateTime.UtcNow - site.LastContentModifiedDate).Days;
    if(daysSinceLastChange > 60)
    {
        site.Delete();
    }
}
```