



Microsoft SharePoint Online
Developer Guide (Beta)

This document is provided "as-is". Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2011 Microsoft. All rights reserved.

Contents

Welcome to the Microsoft SharePoint Online Developer Guide.....	6
What is SharePoint Online?	6
Advantages of SharePoint Online	7
Advantages for Information Workers.....	7
Advantages for Developers.....	7
Advantages for IT Professionals.....	7
Summary of What's New for Developers.....	8
Capabilities and Patterns of SharePoint Online	11
Customization Capabilities and Patterns	11
Customization Limitations	12
Common Types of Development for SharePoint Online	12
Ad-Hoc Customization and Prototypes.....	12
Single Web Part Solution	12
Collaborative Business Solutions.....	13
Customizations to SharePoint Workloads.....	13
Developer Tools.....	14
Customizing SharePoint Online Sites Through the Browser	16
Typical Patterns for Browser-based Customizations and Development.....	16
Lists and Libraries	16
Content Types	17
Site Pages	17
Site Templates	18
Subsites	18
Site Properties	18
Layouts	19
Content Editor Web Part	19

jQuery and the Content Editor Web Part.....	19
Customizing SharePoint Online Sites by Using SharePoint Designer 2010	20
Using SharePoint Designer as a Development Tool	20
Theming and Branding.....	21
Customizing Master Pages	22
Customizing Page Layouts	23
Working with Custom Cascading Style Sheets.....	23
Creating SharePoint Workflows	24
SharePoint Events and Custom Actions	25
Using Web Parts for Views and Forms in SharePoint Designer 2010	25
Building, Testing and Deploying Sandboxed Solutions in SharePoint Online by Using Visual Studio 2010	27
Typical Patterns for Developing Sandboxed Solutions by Using Visual Studio 2010.....	27
Overview of Sandboxed Solutions	27
Sandboxed Solutions and Visual Studio 2010	28
Creating Development and Test Environments	28
Creating Site Collections in SharePoint Online to Validate Deployment.....	28
Visual Studio 2010 SharePoint Power Tools.....	29
Build Process.....	29
Debugging Sandboxed Solutions by Using Visual Studio 2010	29
Outputting Debug Information in SharePoint Online.....	30
Restrictions for Sandboxed Solutions.....	30
Allowed and Disallowed Artifacts.....	30
Allowed and Disallowed Operations.....	32
Exception Handling Characteristics of Sandboxed Solutions	32
Using Remote APIs in SharePoint Online Solutions	32

Typical Patterns for Developing SharePoint Online Solutions That Use the Remote APIs	33
Working with Client-Based APIs for SharePoint 2010.....	33
Client Object Model Architecture.....	34
Client Object Model Processes.....	34
Developing .NET Framework Clients for the SharePoint Client Object Model	34
Synchronicity	35
Developing Silverlight Clients for the SharePoint Client Object Model.....	35
Synchronicity	36
Developing JavaScript Clients for the SharePoint Client Object Model	36
Synchronicity	36
Ribbon Controls and Menu Items	36
Creating Menu Items	36
Creating Ribbon Controls	37
Accessing SharePoint Objects from Custom Actions	37
Creating Client-Side Dialog Boxes	37
Client Authentication in Solutions That Use Remote APIs.....	38
Authentication in the ECMAScript Implementation of the Client Object Model.....	39
Authentication in the Silverlight Implementation of the Client Object Model.....	39
Authentication in the .NET Implementation of the Client Object Model.....	39
SharePoint Online Web Services.....	40
Available SharePoint Online Web Services	40
Conclusion.....	42
Appendix A – Setting Up Your Local Environment for SharePoint Online Solution Development.....	44

Welcome to the Microsoft SharePoint Online Developer Guide

The Microsoft SharePoint Online Developer Guide is designed to help you gain knowledge and understanding of SharePoint Online within Microsoft Office 365 as you build and extend your online sites to meet your business needs.

SharePoint Online provides a solid business collaboration platform on which developers can build solutions rapidly by using familiar development tools. In addition to in-browser customizations, SharePoint Online supports development with Microsoft SharePoint Designer 2010, Microsoft Visual Studio 2010, and Microsoft Visual Studio Team Foundation Server 2010. SharePoint Online enables you to modify sites directly and deploy code as sandboxed solutions that are run in a protected environment to safeguard the environment from poorly performing or malicious code.

SharePoint Online is a major step forward for business collaboration deployment options. At its core, SharePoint has a rich set of features accessible to developers of all skill levels. SharePoint Online takes this same developer toolset and ecosystem and moves it into the cloud within Office 365.

This guide walks you through some of the rich features in SharePoint Online that are available to developers and designers. It provides an overview of the feature set and extensibility points for SharePoint Online, and an understanding of how to create solutions for this new environment. This guide begins by describing the types of solutions you can build, and then addresses the developer tools for SharePoint 2010, the new platform features, and the solution deployment architecture.

What is SharePoint Online?

SharePoint Online is a cloud-based service, hosted by Microsoft, for businesses of all sizes. Instead of installing and deploying Microsoft SharePoint Server 2010 on premises, any business can now simply subscribe to SharePoint Online to provide their employees with an enterprise-grade solution for creating sites to share documents and information with colleagues and customers.

Moving your SharePoint infrastructure to the cloud does not change your ability to customize your environment so that it meets your business needs. Information workers,

designers, and developers can modify the SharePoint Online environment by using many of the same tools and techniques that you would use for an on-premises deployment.

Advantages of SharePoint Online

SharePoint Online provides advantages to various people in your organization, including information workers, developers, and IT professionals.

Advantages for Information Workers

SharePoint Online offers a comprehensive set of functionalities spread across the different workloads, such as sites, social collaboration, search, content management, composites, and business insight. With SharePoint Online, your users can be productive very quickly.

More information about SharePoint workloads:

[SharePoint Workloads and Capabilities](#)

Advantages for Developers

SharePoint Online opens a new arena for developers to create solutions for customers who use the Microsoft Office 365 service instead of maintaining on-premises servers, or for clients who are looking to take advantage of hybrid deployments where some data is stored behind the firewall and other data is off-loaded in the cloud.

Sandboxed solutions, Microsoft Silverlight support, and the new client object model enable powerful custom solutions to be installed in SharePoint Online.

Advantages for IT Professionals

SharePoint Online simplifies IT management by removing the need to deploy, configure, monitor, update, or upgrade a collaboration solution on your premises. You can use the Microsoft Online Services Administration Center to create new sites, install solutions, and provide access to specific users.

SharePoint Online uses redundant and geographically dispersed data centers. Each data center houses a reliable and redundant infrastructure to support SharePoint Online.

By allowing Microsoft to assume much of the operational burden that comes from managing the infrastructure that is associated with on-premises software, businesses can focus their resources on what is important—running their businesses. Small-sized

businesses and medium-sized businesses can now take advantage of the same enterprise-grade technologies that are available to larger-sized companies, without having to shoulder the operational and hardware infrastructure necessary to host SharePoint on-premises.

Summary of What's New for Developers

SharePoint Online includes many new capabilities and features for developers. The following tables summarize what's new for developer productivity and for rich platform services in SharePoint Online.

Developer Productivity	
SharePoint Designer 2010	SharePoint Designer 2010 enables you to rapidly create SharePoint solutions in response to business needs by leveraging the building blocks that are available in SharePoint in an easy-to-use environment. For example, in SharePoint Designer, you can modify and brand the user interface, build workflows, define custom actions, and manipulate SharePoint lists and libraries.
SharePoint development tools in Microsoft Visual Studio 2010	Visual Studio 2010 provides support for SharePoint development that you can use to build applications for your SharePoint Online environment. Support includes SharePoint project templates to help get your development started. Use Visual Studio 2010 to create SharePoint solution package (.wsp) files that enable you to deploy your solutions to your SharePoint Online environment.
Windows 7 and Windows Vista Operating System Support	SharePoint 2010 can be installed on Windows 7 64-bit and Windows Vista 64-bit operating systems for development purposes. This provides you with an opportunity to develop and test solutions for SharePoint Online without requiring that you deploy a server infrastructure on your development computer.

More information about developer productivity:

[Setting Up the Development Environment for SharePoint 2010 on Windows Vista, Windows 7, and Windows Server 2008](#)
[SharePoint Designer 2010](#)

Rich Platform Services	
Sandboxed solutions	<p>A sandboxed solution is deployed at the site collection level by using the SharePoint solution gallery, and it cannot access data outside of the site collection in which it was deployed. In addition, a sandboxed solution cannot invoke certain security-related functionality, such as running with elevated privileges. All of these restrictions are put in place to ensure that malicious or poorly performing code cannot adversely affect the SharePoint Online environment. As a result, site collection administrators can feel confident about uploading and activating a sandboxed solution.</p> <p>SharePoint Online supports the deployment of sandboxed solutions that restricts the operations that your code can perform, and also provides a monitoring environment to verify that code does not adversely impact other sites.</p> <p>Note: Developing a sandboxed solution is the only way you can upload and run custom code in SharePoint Online.</p>
SharePoint Server ribbon	<p>The SharePoint Server ribbon provides a consistent user interface for working with SharePoint objects. The ribbon includes tabs and controls that appear at the top of the browser interface, providing a consistent location to perform tasks for users of your SharePoint site.</p>
SharePoint dialog boxes	<p>SharePoint provides a dialog box platform that enables modal dialog boxes to be included in custom solutions that fit in with the style of a SharePoint page. By using dialog boxes, users are encouraged to focus on the information that is presented to them, and then interact with and close the dialog box before continuing to work with the site. Dialog boxes enable the creation of responsive user interfaces so that users can complete data entry boxes from the pop-up dialog box on the existing page, instead of having to navigate to a new page to complete the data entry task. In many cases, they improve performance by reducing the number of page requests and round trips to the server.</p>
Silverlight Web Part	<p>SharePoint 2010 provides a Silverlight Web Part that you can use to easily add a Silverlight application to a page in your site. You can upload a Silverlight application to a library in your SharePoint site and then configure the Silverlight Web Part to load your Silverlight application.</p>

<p>List lookups and relationships</p>	<p>SharePoint 2010 enables you to create relationships between lists in your SharePoint site collection. For example, you can specify that the column for a list can take values that exist only in another list elsewhere on the site. Using a defined lookup, you can use join statements in Collaborative Application Markup Language (CAML) or LINQ to SharePoint to query across lists.</p>
<p>LINQ to SharePoint</p>	<p>SharePoint 2010 includes the LINQ to SharePoint provider, which translates LINQ statements to CAML queries. LINQ is a standard query language and, when applied to SharePoint, it enables developers to query SharePoint lists without requiring them to learn the CAML syntax that is used internally by SharePoint.</p>
<p>Event enhancements</p>	<p>SharePoint 2010 provides new events that you can use when developing applications for your SharePoint site. Events enable you to intercept an action and run code either pre-event or post-event, as part of a sandboxed solution.</p>
<p>Workflow enhancements</p>	<p>In SharePoint 2010, you can associate workflows with the site and with a particular list. This enables you to create workflows that affect multiple lists or start workflows that do not affect lists.</p> <p>Workflows also support new events that enable complex business processes to be chained together in a modular, manageable way.</p> <p>Note: You can develop code for workflow events only in sandboxed solutions.</p>
<p>Client object model</p>	<p>The client object model is a client-side set of technologies that exposes SharePoint 2010 server-side objects and data on client computers. By using the client object model, you can develop applications that run on the client computer through familiar development concepts, such as objects, properties, events, and methods.</p>
<p>REST APIs</p>	<p>SharePoint 2010 provides access to server data through the Representational State Transfer (REST) API. By using REST, lists and list items are represented as HTTP resources that can be addressed by remote URLs.</p>

More information about rich platform services:

[Developing, Deploying, and Monitoring Sandboxed Solutions in SharePoint 2010](#)

[List Relationships in SharePoint 2010](#)

[New Events in SharePoint Foundation 2010](#)

Capabilities and Patterns of SharePoint Online

SharePoint Online provides many of the features you expect from SharePoint 2010, such as document sharing, collaboration, and workflows. In addition, you can use SharePoint Designer 2010 to design and modify your site's appearance and core collaboration functionality. You can also use Visual Studio 2010 to develop powerful custom solutions for the SharePoint Online environment.

Customization Capabilities and Patterns

You can use the customization capabilities of SharePoint Online to build business solutions that match your specific requirements. The following table lists some of the types of customization options available to you, and describes the patterns and tools you can use to implement those types of customizations.

Customization Capabilities and Patterns	
Create and deploy no-code workflows	Use SharePoint Designer to create and deploy no-code workflows.
Customize content types	Use SharePoint Designer to customize content types.
Brand master pages	Use SharePoint Designer to brand master pages.
Create page layouts	Use SharePoint Designer to create page layouts.
Create and deploy site templates	Use SharePoint Designer to create and deploy site templates.
Consolidate, filter, roll up, and render data	Use the Data View Web Part to create mashups, filtered views, rollups, and general renderings of SharePoint data or data consumed from web services or RSS feeds.
Create browser-based forms	Use Microsoft InfoPath 2010 to design browser-based forms for lists and workflows.
Access and manipulate data from HTTP requests	Use the SharePoint web services to access and manipulate data from HTTP requests without writing code for that manipulation.
Create and deploy custom code-based solutions	Use Visual Studio 2010 to create code-based sandboxed solutions that can be deployed to SharePoint Online environments.

Customization Limitations

Although SharePoint Online supports many of the customization options that are available to a SharePoint 2010 on-premises deployment, some customizations are not supported. For example:

- You cannot deploy pluggable authentication providers, site definitions, or other features and solutions that require deployment and configuration on the server at the farm level.
- You cannot modify built-in SharePoint files, web.config settings, or security policies.
- You cannot make configuration changes that affect the web server settings or the Microsoft .NET Framework.
- You cannot make changes or add capabilities that require a custom database, or that require changes to an existing database.
- You cannot make changes to the underlying Windows Server and Microsoft SQL Server platform layers.

Common Types of Development for SharePoint Online

SharePoint Online provides support for the development of your business applications. Most development projects for SharePoint can be categorized into one of four major project types. These types, and their key features, are described below.

Ad-Hoc Customization and Prototypes

Changes and customizations (such as branding, creating and manipulating lists, and setting site properties) are often performed on live SharePoint Online sites by using the web browser or SharePoint Designer 2010. Additionally, saving a modified site as a template by using these same tools is a common development task. This template is often used as the starting point for further development in Visual Studio.

Single Web Part Solution

Single Web Part solutions can range from a simple Web Part that renders data from a SharePoint list in a specific way, to a complex solution that displays data from several sources and wraps additional functionality around that data. You can build Web Parts by using Visual Studio 2010.

Collaborative Business Solutions

Collaborative business solutions are designed to facilitate and encourage users to work together toward a common goal. These types of solutions typically include multiple SharePoint features, such as lists, forms, workflows and event receivers that can be combined to implement a collaborative solution. For example, you could build a collaborative solution that manages the recruitment process for an organization by combining standard SharePoint features.

You can build collaborative solutions by using the browser, SharePoint Designer 2010, or Visual Studio 2010. The approach that you use depends on the complexity of the solution that you are building.

Customizations to SharePoint Workloads

SharePoint offers six workloads that you can customize by using well-defined extensibility points. The following table briefly describes the workloads.

SharePoint Workloads	
Sites	SharePoint Sites provide a single infrastructure for all your business websites. You can share lists and documents with colleagues and partners, or publish information to customers. The Sites workload provides security features and management that enables you to secure your content and ensure it is targeted to the appropriate users. Sites also support mobile devices and enable users to work with content offline.
Communities	SharePoint Communities use SharePoint Online as a single management platform to deliver various collaboration tools (such as blogs, wikis and people search). Communities provide your users with opportunities to share ideas, find colleagues, and create social content. The Communities workload includes tagging, rating, and the tag cloud, which can all encourage a sense of community among your users.
Composites	SharePoint Composites offer tools and components for you to build your solution by selecting from existing features without requiring you to write code. Make use of built-in features to assemble powerful business solutions. Use the built-in workflows to create approval and review workflows in SharePoint Designer or in Microsoft Visio.

Content	SharePoint Content workloads simplify content management with features such as document types, retention policies, and tight integration with Microsoft Office and SharePoint enterprise search. Use features such as page layouts to add variety to your SharePoint deployment, and meet compliance needs with strict document retention capabilities.
Insights	SharePoint Insights give users access to data stored in business applications through dashboards and scorecards, so they can make decisions based on that data. Using Excel Services, you can easily provide access to Microsoft Excel workbooks stored in SharePoint Online.
Search	SharePoint Search workloads provide discoverability for documents, list data, and people.

There are several general approaches to customizing or extending these core SharePoint workloads. Customize the workloads by using the browser, SharePoint Designer 2010, or Visual Studio 2010. The approach that you use will depend on the complexity of the solution you are building.

More information about the SharePoint workloads:

- [SharePoint Sites](#)
- [SharePoint Communities](#)
- [SharePoint Composites](#)
- [SharePoint Content](#)
- [SharePoint Insights](#)
- [SharePoint Search](#)

Developer Tools

You can use various tools to build solutions targeted at SharePoint Online. Tools include the browser, SharePoint Designer 2010, and Visual Studio 2010.

In addition, SharePoint provides a .wsp file packaging format for SharePoint solutions that enables you to share solutions between tools such as SharePoint Designer 2010 and Visual Studio 2010. This common packaging format simplifies the transition between the tools and enables collaboration among users with different skills. For example, a solution designed in SharePoint Designer 2010 by a site designer can be easily packaged and provided to a developer in the standard .wsp format as a starting point for more development in Visual Studio 2010.

The following table summarizes the tools for building solutions for SharePoint Online that are available to developers.

Note: Later sections in this developer guide describe how to use the tools in this table to create solutions and customizations for SharePoint Online.

Developer Tools	
Browser	<p>SharePoint 2010 provides options for you to customize the SharePoint site by using functionality available through the SharePoint browser interface. You can easily switch pages into edit mode and then add or remove Web Parts, content, and images in the user interface. Using the browser, you can also make other changes to your SharePoint site, such as adding or configuring lists, content types, or workflows, and changing the configuration of the site from the Site Settings page. You can also use the browser to change the theme of the site. By selecting a theme, you can change the appearance of all pages in your site from one place.</p>
SharePoint Designer 2010	<p>SharePoint Designer 2010 is an important tool in the solution creation lifecycle for SharePoint Online. You can create and configure sites by using SharePoint Designer 2010 and you can package them into a SharePoint solution (.wsp) file that can be exported for more modification by Visual Studio 2010.</p> <p>SharePoint Designer 2010 has been designed around the artifacts that you create in SharePoint, such as lists and libraries, workflows, content types, data sources, site level settings, master pages, and page layouts.</p>
Visual Studio 2010	<p>Visual Studio 2010 includes support for the most common types of projects that you might want to build with SharePoint Online. You should note that Visual Studio supports two forms of SharePoint 2010 solutions: farm and sandboxed.</p> <p>Farm solutions are registered in the global assembly cache and run under full trust. Farm solutions are not supported by SharePoint Online.</p> <p>Sandboxed solutions are deployed to the site collections solution gallery and run in a restricted execution environment. Sandboxed solutions <i>are</i> supported in SharePoint Online. By creating sandboxed solutions in Visual Studio 2010, you can extend the capabilities of your SharePoint Online solution.</p> <p>A number of project templates for creating SharePoint items are provided with Visual Studio, and these templates are available in either Microsoft</p>

Customizing SharePoint Online Sites Through the Browser

This section drills down into how to customize SharePoint Online sites by using a browser.

Typical Patterns for Browser-based Customizations and Development

Using the browser to make customizations is appropriate when you are making *ad-hoc* changes to your solution, or when you are creating objects that will be used as part of a site template.

Lists and Libraries

You can create lists and libraries in your SharePoint Online site by using the browser. When you create a SharePoint list, you can pick an existing list template or you can create a custom list that starts with a basic template. After you create the list, you can add or remove columns so that it meets the needs of your application. By using the browser to create your lists and libraries, you can more rapidly respond to the *ad-hoc* storage requirements of your organization.

You can also specify lookup columns to display data from existing lists in your new list. For example, you can create a list of office locations and use links to that list from other lists that you create in your solution.

As with lists, when you create a new library for your SharePoint site, you can specify an existing template on which to base the new library. For example, you can create a basic document storage library, or you can create a library to store pictures, slides, forms and other content.

Lists and libraries automatically generate their own user interfaces, such as Web Parts, forms, and dialog boxes for working with the files and list data. However, if your requirements include the creation of custom interface components or workflows for specific lists or libraries, it may be more appropriate to create the lists and libraries in a package and deploy them with the other components from Visual Studio 2010.

Packaging and deploying lists and libraries with other components is discussed later in this Developer Guide.

More information about creating lists and libraries by using the browser:

[Create Lists with Custom Content Types in SharePoint 2010](#)

[Create Linked Lists in SharePoint 2010](#)

Content Types

A content type is a reusable collection of columns, workflows, behaviors and other settings for a category of items or documents in your SharePoint Online site. For example, you can create a content type that represents an expense request and include the information that must be captured and a workflow that must be followed when an individual creates a new expense item.

You can create new content types by using the browser by accessing the Site Settings page from the **Site Actions** menu, and then selecting **Site content types** in the **Galleries** section. You can modify the new content type by adding columns to it and by specifying other properties that define how information that is associated with this content type will be managed.

The creation of a new content type is often associated with other activities, such as adding corresponding workflows, lists, or libraries that use the content type you have defined. Each of these tasks can also be performed by using the browser. However, more complex requirements are often defined and deployed together as a SharePoint Feature. To define a SharePoint Feature, you should use Visual Studio 2010.

More information about creating content types by using the browser:

[Introduction to Content Types](#)

[Creating Content Types](#)

Site Pages

The browser provides an easy-to-use interface for creating new pages in your site. By using the ribbon or the **Site Actions** menu, you can switch to edit mode and use the in-place editing features of SharePoint Online to customize pages so that they meet your requirements.

The ribbon provides a rich editing interface and the option to change your page layout and add new content, such as text, images, and Web Parts. You can also create pages by

using the browser and export those pages as part of solution packages for inclusion in projects that require additional development.

More information about creating site pages by using the browser:

[Web Authoring in SharePoint 2010](#)

Site Templates

Site templates provide a starting point for you to create a site with a preconfigured structure, features, and content. You can use the browser to easily create a new site from a template, and you can create a new template in the browser by saving an existing site as a template from the Site Settings page. When you save your current site as a template, you can specify the file name, template name, and description, and you can choose to include the current site content in the template file. Newly created templates are located in the solution gallery and appear on the new site dialog box when you choose to create a new site.

You can export site templates that you create by clicking the corresponding .wsp file in the solution gallery. You can then modify them further by using Visual Studio 2010.

More information about creating site templates by using the browser:

[A preview of the SharePoint Server 2010 site templates](#)

[Save a SharePoint site as a template](#)

[SharePoint 2010 and web templates](#)

Subsites

You can use the browser to create subsites in your site collection by clicking **New Site** on the **Site Actions** menu. You can select site templates from the list of site templates that appears. You can specify the name and address for the subsite and choose to use the same permissions that are set for the parent site, or you can define unique permissions for the subsite.

Site Properties

The browser provides an easy way to modify the properties of your site, and you can change settings and view those changes immediately. To modify the site's properties, on the Site Settings page, select the options you want to change from the corresponding category. For example, you can change the basic properties of the site, such as the title,

description and icon, and you can also configure the menu settings and add your own menu items to the SharePoint interface.

Layouts

SharePoint provides a flexible layout system for pages that you create on your site. For example, you can specify that the page content is split into several columns with optional headers, footers, and sidebars to get a page layout that is suitable for your application. You can use the browser to pick a layout for a page by selecting edit mode and then choosing a text layout from a list of options.

Using the browser is appropriate if the layout you require is supported by the built-in layouts. If that is not the case, you may have to use SharePoint Designer to create the page layout that you want.

Content Editor Web Part

The Content Editor Web Part is a simple Web Part that enables you to add markup directly to a SharePoint page. Despite its simplicity, the Content Editor Web Part is very powerful and can contain HTML, ECMAScript (JavaScript, JScript), and CSS. For example, you can add a Content Editor Web Part to your page and then add CSS markup to style other elements on the same page.

Adding the Content Editor Web Part to your page enables you to control the content of the page, and provides you with the ability to add HTML content and CSS styling from the browser. This is an appropriate approach for changes to specific pages on your site.

More information about using the Content Editor Web Part:

[About the Content Editor Web Part](#)

[Content Editor Web Part](#)

[Make your SharePoint pages pop with the Content Editor Web Part](#)

jQuery and the Content Editor Web Part

jQuery is a fast and concise JavaScript library that you can use in your SharePoint Online deployment. jQuery enables you to find and manipulate HTML elements with minimal JavaScript code. jQuery code is executed in the browser.

jQuery makes JavaScript code easier and quicker to write. The library provides helper functions that dramatically increase your productivity. The resulting code is easier to

read and more robust because the high level of abstraction hides many of the checks and error handling procedures. The library supplies a powerful interface for selecting DOM elements that goes far beyond the simple search for elements that match a given ID. For example, you can easily select:

- All elements that share a given CSS class.
- Those that have certain attributes.
- Those that appear in a given position in the document.
- Those have a relationship to other elements.

More importantly, you can add filter conditions and you can chain all of these query features together to meet your specific requirements. To integrate jQuery into SharePoint Online, you can upload the API into a document library, and then add a Content Editor Web Part that references and uses that API in your SharePoint Online sites.

More information about using jQuery and the Content Editor Web Part:

[jQuery Web Site](#)

[Explore Rich Client Scripting With jQuery, Part 1](#)

[Explore Rich Client Scripting With jQuery, Part 2](#)

[Using jQuery with SharePoint 2010](#)

[Adding JavaScript Tabs to SharePoint](#)

[Two examples of using SharePoint & jQuery](#)

Customizing SharePoint Online Sites by Using SharePoint Designer 2010

This section examines in detail using SharePoint Designer 2010 to customize SharePoint Online sites.

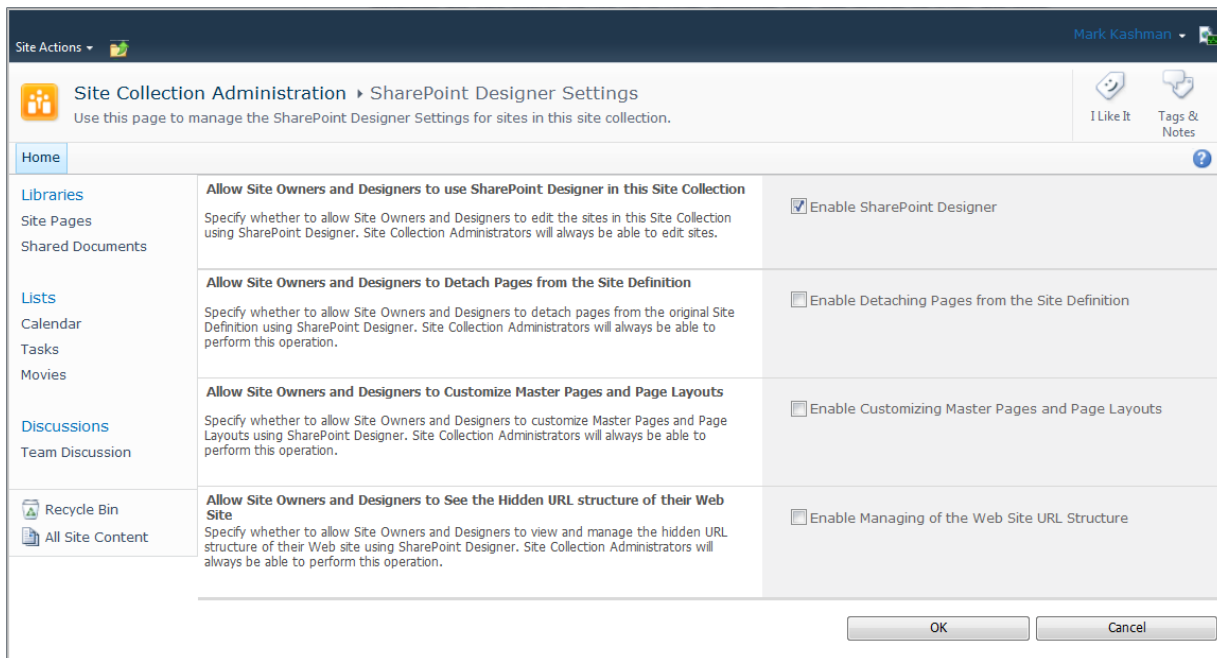
Using SharePoint Designer as a Development Tool

Using the browser to make customizations is appropriate when you are making *ad-hoc* changes to your solution or when you are creating objects that will be used as part of a site template.

SharePoint Designer 2010 is an ideal choice for rapid development of SharePoint applications for advanced users, site designers, and developers. By using SharePoint

Designer 2010, you can construct no-code solutions from the building blocks available in SharePoint 2010. In addition, you can use SharePoint Designer to get a quick start on your SharePoint development projects.

The ability to modify your SharePoint Online site by using SharePoint Designer 2010 should be restricted to particular users, and SharePoint Online provides the site collection administrator with the option to enable or disable various SharePoint Designer features. These settings are configured from the SharePoint Designer Settings page that is available from the Site Settings page. The following figure shows the available options.



More information about getting started with SharePoint Designer:
[Customizing SharePoint Online Using SharePoint Designer 2010](#)

Theming and Branding

SharePoint Designer 2010 makes it easy to change the theme and branding of your SharePoint site. By using SharePoint Designer 2010 to modify your site, you can incorporate changes that are not available to you through the Site Settings page in the SharePoint browser interface. By using SharePoint Designer 2010, you can add a company logo, select a color scheme, and modify the headers and footers for your

SharePoint Online site collection. This branding is then inherited by the site collection's subsites.

Note: When you select Change site theme in SharePoint Designer 2010 you are directed to the Site Theme page on the SharePoint site, where you can change the theme by using the browser.

Customizing Master Pages

Master pages are a Microsoft ASP.NET feature that is used in SharePoint 2010 to specify a consistent design of the site. This consistent design is inherited by content pages that use the master pages. Master pages make development of your site much easier because when you create a new content page based on a master page, you do not have to repeat the markup that you defined in the master page. When a content page is rendered, the content page is merged with the master page to provide the output that is presented to users.

A master page is an ASP.NET file that has the extension .master and that has a predefined layout of HTML elements and controls. The master page has a specific directive that identifies the page as a master page and it also includes the basic HTML structure for your pages. For example, you can include a logo and menu structure on your master page, and a footer that you want to appear on all of the pages in your site. The master page uses content placeholders to specify the location in which to display content on the page, and an identifier that is used to reference the location when a content page is loaded.

In addition to providing a layout and other elements that you can share across multiple pages in a SharePoint Online site, master pages can also contain CSS and ECMAScript (JavaScript, JScript) references that define the overall look, feel, and functionality of your site. If required, you can use a different master page for various sites in your site hierarchy to distinguish the areas of your portal.

You can edit a master page from your SharePoint site by using SharePoint Designer 2010 to open and edit the page. You can also create new master pages by using SharePoint Designer, but a master page that you create using this approach will not have all of the content placeholders that SharePoint 2010 requires to function correctly. However, you can add the required placeholders to your page manually, or you can copy the contents of an existing starter master page to help get you begin.

More information about working with master pages in SharePoint Designer:[ASP.NET Master Pages](#)[Working with SharePoint 2010 Master Pages](#)[Real World Branding with SharePoint 2010 Publishing Sites](#)[Starter Master Page](#)[Starter Master Pages for SP2010](#)

Customizing Page Layouts

Page layouts are an important feature in SharePoint publishing sites that define how a page should look, and the content that should be present on the page (such as lists and libraries). You can edit page layouts in SharePoint Designer 2010 by selecting the page that you want to edit from the page layouts object collection in the SharePoint Designer navigation folder.

After you open a page layout for editing in SharePoint Designer 2010, you can edit it just as you would edit any other page that is attached to a master page. When you add custom content to a page layout, the content appears on all the publishing pages that are based on that page layout.

For example, you can use any of the following elements in page layouts:

- **Tables.** Create tables to align content.
- **Pictures.** Click and drag images from the Images folder to a section of the page.
- **Links.** Insert hyperlinks or bookmarks.
- **Data views and forms.** Insert a Data View Web Part or a form.
- **Controls.** Insert HTML controls, ASP.NET controls, SharePoint controls, or Data Source controls.
- **Web Parts.** Insert a Web Part and give visitors to the page the option to add or remove Web Parts (such as views of document libraries and lists).
- **Symbols.** Insert symbols that are not available from your keyboard.

More information about working with page layouts in SharePoint Designer:[Page Layouts Model](#)[Page Layouts and Master Pages](#)

Working with Custom Cascading Style Sheets

SharePoint Designer provides an ideal interface for you to work with the cascading style sheets in your site. When you edit the style sheet on the master page, your changes are

applied to all of the child pages that use the master page. Before editing the cascading style sheets in your site, you should consider making a copy of the master page.

When you are using SharePoint Online, you will not have access to the underlying folder location to upload or modify the cascading style sheets that are provided with SharePoint 2010. Therefore, you should make your changes to a new style sheet and upload that new style sheet to a library on the server. You can then reference the new style sheet from the master page.

More information about working with cascading style sheets in SharePoint Designer:

[Working with the SharePoint Theming Engine](#)

Creating SharePoint Workflows

You can design workflows for your SharePoint deployment by using the graphical modeling interface, or workflow designer, that is available in SharePoint Designer 2010. The Workflow Designer enables you to specify a set of steps, conditions, and actions that fit together into a workflow without requiring you to write code. Workflows can also be designed in Visio 2010 and imported into SharePoint Designer 2010.

The Workflow Designer in SharePoint Designer 2010 is used to create workflows on the currently opened SharePoint site. You can create three types of workflows by using SharePoint Designer:

- **List workflows.** Using this mechanism, you attach the workflow directly to a list or library on the site. Use the List workflow when you are making a workflow that is specific to a list or library and does not have to be deployed for use on a different list or library.
- **Reusable workflows.** This type of workflow is created with reusability in mind. Create a reusable workflow when you intend to attach it to a content type and use that content type in a list or library.
- **Site workflow.** Site workflows are not attached to a list or library. They work on the site itself. Use this workflow if you do not want to restrict the automated process to a list or library on the site. For example, you can use the site workflow to take a survey of the site members or to execute a process on a Document Set.

You do not need to create all workflows from scratch. The built-in workflow templates (**Approval**, **Collect Feedback**, and **Collect Signatures**) that can be used in the browser can also be extended by using the Workflow Designer. These workflows are categorized

as **Globally Reusable Workflows** and are visible and available to every site in the site collection.

SharePoint Events and Custom Actions

In addition to creating workflows with SharePoint Designer 2010, you can also develop and deploy powerful workflow artifacts, such as custom workflow actions and server-side workflow event handlers. You do this by including them in Visual Studio-based sandboxed solutions. For more information about developing sandboxed solutions, see [Building, Testing and Deploying Sandboxed Solutions in SharePoint Online by Using Visual Studio 2010](#) later in this Developer Guide.

More information about creating and editing workflows in SharePoint Designer:

[SharePoint Designer Workflow Overview](#)

[Workflow Development in SharePoint Designer](#)

[Workflow Development for SharePoint Foundation](#)

[Workflow Development Tools Comparison](#)

[SharePoint Designer's New Workflow Editor: Introduction](#)

[SharePoint Designer's New Workflow Editor: Keyboard Shortcuts](#)

[Overview of Human Workflow in SharePoint 2010](#)

Using Web Parts for Views and Forms in SharePoint Designer 2010

In SharePoint Designer 2010, you can create list views and data views and list forms and data forms. You create such views and forms to help site visitors interact with the data on the site, and in the case of a custom solution, to manage how data is presented and modified. Every view and form you create in SharePoint Designer 2010 is stored in a Web Part. The Web Part contains the code needed to render the view or form on the page and enable users to change the appearance of the data or manipulate data in the list. Depending on which view or form you choose and the type of data source you are using, a different Web Part is added to the ASPX page. Some of the Web Parts are editable in SharePoint Designer 2010, some in the browser, and some require related SharePoint tools such as InfoPath 2010.

The following table briefly describes the various Web Parts that are used for the views and forms you create in SharePoint Designer 2010.

Data View Web Parts

Data View Web Parts	
XSLT List View Web Part	The standard and often default view associated with lists and libraries. This Web Part is also used when you create a view on a page where the data source is a list or library. This Web Part has full ribbon support in SharePoint, so the view can be customized in the browser or in SharePoint Designer.
Data Form Web Part	Used by default when you create a data view to a data source other than a list and library. You can use the Data Form Web Part for lists and libraries when the Data Source Details pane is used, and it can be used as a view or a form to any data source, including lists, libraries, and web services.
Data View Web Part	Used in the previous releases of SharePoint, and is used only in certain upgrade scenarios in SharePoint 2010. It is unlikely that you will use this Web Part in your SharePoint Online solutions.
List Form Web Part	The default list form used for SharePoint lists and libraries. This Web Part provides a well-formatted, easy-to-use form for users. To customize the form, however, you must replace it with a Data Form Web Part or upsize the forms to InfoPath.
List View Web Part	Used for highly specialized views, like the Calendar view, Gantt view, and Datasheet view. The Web Part is well-formatted and provides powerful views of the list or library. To customize the views shown in the Web Part, you can use the SharePoint Designer code view.
InfoPath Form Web Part	Used to host InfoPath-based list forms. The Web Part is created when you customize list forms in InfoPath 2010. It is highly customizable and renders powerful forms generated by InfoPath 2010 form templates. This Web Part cannot be customized directly in SharePoint Designer, but the integration between SharePoint Designer and InfoPath 2010 enables you to launch the InfoPath 2010 editor from the SharePoint Designer 2010 environment.

More information about creating and managing data views and forms:

[XSLT List View Web Part](#)

[Data Form Web Part](#)

[Data View Web Part](#)

[List Form Web Part](#)

[List View Web Part](#)

[InfoPath Form Web Part](#)

Building, Testing and Deploying Sandboxed Solutions in SharePoint Online by Using Visual Studio 2010

Sandboxed solutions are a new feature of SharePoint 2010. Sandboxed solutions provide many benefits for information workers, site administrators, farm administrators, and the SharePoint Online environment.

Typical Patterns for Developing Sandboxed Solutions by Using Visual Studio 2010

As a developer, you can use Visual Studio 2010 to create sandboxed solutions. You will typically create sandboxed solutions when you need to:

- Deploy a solution that contains functionality (such as server-side code) that cannot be achieved by using SharePoint Designer 2010 or the browser.
- Create a reusable deployment package that will be installed in multiple SharePoint Online sites.
- Create an ISV solution for your customers who use SharePoint Online.

Overview of Sandboxed Solutions

Sandboxed solutions are those solutions that are uploaded by administrators to the solution gallery in a SharePoint site collection. In a SharePoint Online environment, it is essential that sandboxed solutions do not have undesirable effects on other site collections, and that they can be monitored and managed easily. The benefits of sandboxed solutions include:

- **Rapid configuration.** Site collection administrators can deploy and manage sandboxed solutions, so that farm administrators do not need to assess, deploy, and manage all the functionality required by information workers.

- **Flexibility.** Sandboxed solutions are run in a separate process that can be restricted by quotas, and their effect on the farm can be monitored.
- **Stability.** Sandboxed solutions can be added to SharePoint sites without the risk of affecting processes outside the sandbox.

Sandboxed Solutions and Visual Studio 2010

When you create a new SharePoint project by using Visual Studio 2010, the SharePoint Customization wizard prompts you to choose a **Farm Solution** or a **Sandboxed Solution**. If you choose **Farm Solution**, and need to change the type to a sandboxed solution after you have created it, you can set the **Sandboxed** property to **true**.

Creating Development and Test Environments

Currently, you cannot deploy solutions directly from Visual Studio to SharePoint Online. Furthermore, you cannot currently attach the Visual Studio 2010 debugger to solutions that are deployed in SharePoint Online. Therefore, you must set up a local development and test environment before you start to create sandboxed solutions for SharePoint Online.

Your goal in creating development and test environments should be to mirror the SharePoint Online environment as closely as possible. For example, you should include SharePoint Foundation 2010 in your local environment, and you should ensure the user code services are running so you can deploy and test sandboxed solutions.

For more information about setting up your development and test environments, refer to [Appendix A – Setting Up Your Local Environment for SharePoint Online Solution Development](#) later in this Developer Guide.

Creating Site Collections in SharePoint Online to Validate Deployment

In addition to setting up your local environment to mirror the SharePoint Online environment as closely as possible, you can also create site collections in SharePoint Online that mirror your production site collections but that are not used by your information workers. This approach enables you to perform final tests and verifications for your solutions before you deploy them to the production site collections. This approach also enables you to output debug information from your solutions in a real SharePoint Online environment, as described in [Outputting Debug Information in SharePoint Online](#) later in this Developer Guide.

Visual Studio 2010 SharePoint Power Tools

The Visual Studio 2010 SharePoint Power Tools include many features that make your development process easier and enable you to concentrate on fulfilling your business requirements. Additionally, the Visual Studio 2010 SharePoint Power Tools are sandbox-aware, and provide you with compile-time checking to ensure you use namespaces and classes that are supported only in sandboxed solutions. Furthermore, the Visual Studio 2010 SharePoint Power Tools enable you to deploy artifacts in sandboxed solutions that would otherwise be disallowed. For example, you can create and package sandboxed versions of Visual Web Parts by using the Visual Studio 2010 SharePoint Power Tools, which otherwise would be disallowed. For more information, see [Restrictions for Sandboxed Solutions](#).

You can download the [Visual Studio 2010 SharePoint Power Tools](#) from MSDN.

Build Process

If you do use restricted namespaces and object types, the build process will still succeed. This is because compilation is performed by Visual Studio against the full object model, regardless of whether the solution is sandboxed. However, when you attempt to deploy a sandboxed solution to SharePoint Online by uploading it and activating it in the solution gallery, SharePoint Online validates the contents of the solution package (.wsp file) and does not deploy it if it contains prohibited artifacts.

When you have tested and debugged your sandboxed solutions, you can use Visual Studio to package them into .wsp files, just as you do for farm solutions. The key difference, however, is that site collection administrators can then simply upload the solution package to the solution gallery and then activate it, instead of requiring farm administrators to deploy the solution.

Debugging Sandboxed Solutions by Using Visual Studio 2010

As previously discussed, you cannot attach the Visual Studio debugger to SharePoint Online server-side processes. So you must debug your solutions in your local development and test environments.

When you press F5 (or use the equivalent menu or toolbar commands) in a sandboxed solution, Visual Studio deploys the solution to the solution gallery in your local SharePoint site collection, and automatically attaches to the **SPUCWorkerProcess.exe**

process. This means that breakpoints, watches, stepping through code, and other debugging features will work for your solution. If you want to debug a solution that is already deployed in your local development or test environment, you can either retract and then redeploy the solution as described previously, or you can attach the debugger to the **SPUCWorkerProcess.exe** process manually.

When you have debugged your solutions in your local development or test environment, you should further test that they run as expected in SharePoint Online by uploading and activating them in your test site collections.

Outputting Debug Information in SharePoint Online

You can further validate the operations being performed in the actual SharePoint Online environment by conditionally creating list items in SharePoint lists, based on any information that you want to interrogate at run time. In effect, you can use SharePoint lists as a custom application log that can help you analyze the operations, exceptions, and performance of your solutions.

If you take this approach, you should consider the implications of creating a large number of list items on your SharePoint Online storage quotas. For example, a good pattern is to conditionally create list items for your data only in the DEBUG configuration of your solution, and to deploy the DEBUG build only to your test site collections. Then, when you are satisfied that your solution is functioning correctly, you can deploy the RELEASE configuration to your production site collections, and you will not use storage quotas needlessly in that environment.

Restrictions for Sandboxed Solutions

To ensure stability for the rest of the farm, some artifacts and operations are not allowed in sandboxed solutions.

Allowed and Disallowed Artifacts

Not all types of Visual Studio projects for SharePoint can be configured as sandboxed solutions, regardless of the code operations that they perform. In general, project types that affect only one site and that do not deploy files to the SharePoint file system are allowed.

Examples of allowed project types include:

- Empty projects (to which you add only allowed project item types, such as Web Parts)
- List Definition projects
- Event Receiver projects
- Content Type projects

Examples of prohibited project types include:

- Site Definition projects
- Business Data Connectivity service (BDC) model projects

In addition, not all project item types can be deployed as part of sandboxed solutions. As with the project types, artifacts that affect only one site and that do not deploy files to the SharePoint file system are allowed, whereas those that affect more than one site or that deploy files to the SharePoint file system, are not allowed. Examples of project item types that can be deployed as part of a sandbox solution include:

- Web Parts
- List definitions
- Event receivers
- Content types
- Visual Web Parts (only the sandboxed versions that you can create with the Visual Studio 2010 SharePoint Power Tools)

Examples of prohibited project item types include:

- Visual Web Parts (non-sandboxed versions that you create without using the Visual Studio 2010 SharePoint Power Tools)
- Application pages
- BDC models

Additionally, access to SharePoint data in a sandboxed solution is restricted to the site collection in which the solution is activated and running. As an example, it is prohibited to instantiate an **SPSite** object with a remote SharePoint URL as its constructor. As another example, it is prohibited to use the **Create** method of the **HTTPWebRequest** class to create a connection to other external websites.

Access to non-SharePoint databases is also prohibited in sandboxed solutions. You cannot include BDC models in your sandboxed solutions, and you cannot create SQL connections.

Allowed and Disallowed Operations

Most objects and operations that affect only the site in which a sandboxed solution is running are generally allowed. For example, your code in a sandboxed solution can access lists, respond to events, create libraries, and render data in Web Parts. However, certain operations at the site level, such as performing these tasks with elevated privileges, are prohibited.

Objects that have a larger scope than the current site cannot be used in sandboxed solutions. For example, you cannot work with **SPFarm** objects or **SPService** objects in your sandboxed solutions. For a full list of allowed objects and operations, see [Microsoft.SharePoint.dll APIs Available from Sandboxed Solutions](#).

Exception Handling Characteristics of Sandboxed Solutions

If you attempt to perform a prohibited operation in a sandboxed solution, exceptions are raised by the **SPUCWorkerProcess.exe** process. Some of these exceptions can be caught and handled by your code, whereas others are handled by the sandboxed environment before your exception-handling code is invoked.

More information about creating and deploying sandboxed solutions:

[Sandboxed Solutions](#)

[Sandboxed Solutions Architecture](#)

[Installing, Uninstalling, and Upgrading Sandboxed Solutions](#)

[Best Practices for Developing Sandboxed Solutions](#)

[Visual Studio 2010 SharePoint Power Tools](#)

[What Can Be Implemented in a Sandboxed Solution?](#)

[Restrictions on Sandboxed Solutions](#)

[Code Sample: Sandboxed Solution Employee Browser](#)

Using Remote APIs in SharePoint Online Solutions

With the SharePoint 2010 client object model, you can connect to and manipulate SharePoint objects and data from applications that run remotely from SharePoint servers.

With previous versions of SharePoint, remote applications that did not run on SharePoint could still consume SharePoint objects and data by using the web services provided by earlier versions of the product. However, those web services were not

© 2011 Microsoft. All rights reserved.

www.microsoft.com/sharepoint

always easy to use, especially when used from some types of applications (such as Silverlight and JavaScript clients), and they did not offer the same power and flexibility provided by the server-side object model.

Now, however, SharePoint 2010 provides a client object model that is as easy to use as the server-side object model is, and it supports many more operations and features than do the traditional web services. You can, of course, still use the web services if they meet your business and technical requirements.

In this section, you will learn about the SharePoint 2010 client object model, and you will also learn about the support for the traditional web services in SharePoint Online solutions.

Typical Patterns for Developing SharePoint Online Solutions That Use the Remote APIs

In brief, you can build any of the following types of applications that use the SharePoint 2010 client object model to access, retrieve, and manipulate SharePoint objects and data:

- Windows-based applications (including Windows Presentation Foundation solutions)
- Console applications
- ASP.NET applications
- Silverlight applications that are hosted in SharePoint Online websites
- Silverlight applications that are hosted in non-SharePoint websites
- Silverlight out-of-browser applications
- JavaScript clients (such as ribbon controls and client-side dialog boxes)

Working with Client-Based APIs for SharePoint 2010

Before you start working with the SharePoint 2010 client object model, you should be aware of the overall architecture and processes for creating remote applications for SharePoint objects and data. You should also be aware of some differences between implementing remote applications for Microsoft .NET Framework applications, Microsoft Silverlight applications, and clients that use JavaScript for communicating with the SharePoint 2010 client object model.

Client Object Model Architecture

The SharePoint 2010 client object model is a client-side set of technologies that exposes site-level data and objects on client computers. The server-side data platform exposes farm settings, sites and webs, native list data, and external list data to server-side applications. The SharePoint 2010 client object model provides a way to access a subset of this functionality, such as sites and webs, and native list data on client computers.

Client Object Model Processes

The SharePoint 2010 client object model presents you with familiar concepts, such as objects, properties, events, type enumerations, and methods that you can use to develop SharePoint solutions. For example, the SharePoint 2010 client object model provides **Web** objects, **Site** objects, and **List** objects, and you can work with these objects in a very similar way to how you develop solutions by using the server-side object model.

However, when you create and manipulate objects, set properties, and call methods of the SharePoint 2010 client object model, those commands are batched on the client and are not executed immediately on the server. Instead, they are batched and formatted as XML description of the operations to perform. Then, when you call the **ExecuteQuery** method or **ExecuteQueryAsync** method, the commands are sent as an XML packet to the **client.svc** service on a SharePoint web server.

The **client.svc** service interprets the XML packet and performs the sequence of commands that it contains. The service then sends back results (or errors) to the calling application in JavaScript Object Notation (JSON) format, where they are presented to your solution again as SharePoint objects and properties.

Developing .NET Framework Clients for the SharePoint Client Object Model

If you want to develop .NET Framework applications that use the SharePoint 2010 client object model, you must reference the following DLLs in your project:

- Microsoft.SharePoint.Client.dll
- Microsoft.SharePoint.Client.Runtime.dll

These DLLs are located in the following path **C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\ISAPI.**

With these DLLs, you can develop remote applications that use the SharePoint 2010 client object model if you are developing them on a SharePoint development server. If you are developing on a non-SharePoint server, you can copy the DLLs locally and reference them on your development computer.

When you distribute your solution to users, you must ensure that the DLLs are included with your application. The easiest way to do this is to create a setup project by using Visual Studio 2010.

Synchronicity

When you develop .NET Framework applications, you can choose between synchronous and asynchronous execution of your commands. That is, you can call either the **ExecuteQuery** method or **ExecuteQueryAsync** method, depending on the requirements of your solution.

More information about developing .NET Framework clients for the SharePoint client object model:

[What's New: Client Object Model](#)

[Using the SharePoint Foundation 2010 Managed Client Object Model](#)

[Managed Client Object Model](#)

[Client Object Model Resource Center | SharePoint 2010](#)

Developing Silverlight Clients for the SharePoint Client Object Model

If you want to develop Silverlight applications that use the SharePoint 2010 client object model, you must reference the following DLLs in your Silverlight project:

- Microsoft.SharePoint.Client.Silverlight.dll
- Microsoft.SharePoint.Client.Runtime.Silverlight.dll

These DLLs are located in the path **C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\TEMPLATE\LAYOUTS\ClientBin**.

These DLLs are built specifically for Silverlight applications—you cannot use the DLLs in the ISAPI folder in Silverlight projects.

When you build your Silverlight solution, the DLLs are automatically included in the solution's XAP file, so you do not need to distribute them separately.

Synchronicity

When you develop Silverlight applications, you can choose between synchronous and asynchronous execution of your commands. That is, you can call either the **ExecuteQuery** method or **ExecuteQueryAsync** method, depending on the requirements of your solution. However, if you are executing commands on the main UI thread, synchronous calls are not allowed—you must use the asynchronous pattern.

More information about developing Silverlight clients for the SharePoint client object model:

[Silverlight Client Object Model](#)

[Module 8: Creating Silverlight User Interfaces for SharePoint 2010 Solutions](#)

[Integrating Custom Silverlight 4 Applications with SharePoint Server 2010](#)

Developing JavaScript Clients for the SharePoint Client Object Model

SharePoint Online provides new user interface elements that information workers can use to perform their tasks efficiently. For example, information workers can use the context-sensitive Server ribbon to access operations when they need to, and they can use the new dialog boxes provided by the client-side dialog box platform to interact effectively with SharePoint data and objects.

As a developer, you can create ribbon controls for the Server ribbon and you can create dialog boxes for the client-side dialog box platform so that information workers can interact with your solutions easily.

Synchronicity

You can use the JavaScript implementation of the SharePoint 2010 client object model in ribbon and dialog box development. The JavaScript implementation of the SharePoint 2010 client object model uses asynchronous execution for responsive user interfaces.

Ribbon Controls and Menu Items

SharePoint menu items and ribbon controls are implemented as objects known as custom actions. Custom actions are defined as elements in XML files, and you can create them by using Visual Studio 2010.

Creating Menu Items

Menu items are custom actions incorporated into the built-in SharePoint 2010 user interface. As examples, the **Site Actions** menu is composed from several built-in, context-sensitive actions that appear as menu item links; and the Site Settings page

includes links that are defined by built-in actions. You can build your own custom actions by creating XML definitions in element files in Visual Studio projects. And you can specify where they appear, such as on the **Site Settings** menu or on a specific administration page.

Creating Ribbon Controls

Ribbon controls are also actions that appear on the SharePoint 2010 Server ribbon. Like menus, the ribbon is context-sensitive and shows appropriate actions to users when users need them. For example, when a user browses to a document library, the ribbon controls for adding new folders and documents are shown. SharePoint 2010 includes many ribbon controls, and groups them together into related actions. For example, the **New Document**, **New Folder**, and **Upload Document** ribbon controls are grouped together on the ribbon.

You can build your own context-sensitive ribbon controls, and you can specify a context in which to show them and a group in which to display them. Like menu items, ribbon controls are custom actions defined in XML element files. As with custom menu items, you develop ribbon controls by creating a new element based on the SharePoint 2010 **Empty Element** project item template, and you add elements to describe the appearance and behavior of the ribbon control.

Accessing SharePoint Objects from Custom Actions

Ribbon code and custom action code run in the user's browser. This means that, normally, this code does not have direct access to the server-side object model. However, custom actions can interact with SharePoint by using the JavaScript implementation of the SharePoint 2010 client object model.

Furthermore, common scenarios for ribbon and menu items include showing client-side dialog boxes that enable users to interact with SharePoint data. Typically, it is the dialog boxes that contain SharePoint 2010 client object model code for interacting with SharePoint, so the custom actions are often just a way of starting the interaction, and may not need to include SharePoint 2010 client object model code.

Creating Client-Side Dialog Boxes

SharePoint Online provides a new client-side dialog box platform that enables information workers to work with SharePoint objects and data efficiently. You can create

your own dialog boxes that enable information workers to interact with your solutions by using Visual Studio 2010.

Many operations in SharePoint 2010 sites are now performed by built-in client-side dialog boxes. For example, the **New** and **Edit** forms for list data are rendered as modal client-side dialog boxes with an HTML interface.

Client dialog boxes run in the user's browser, so the operations a user performs are responsive. Also, these operations do not place heavy loads on the SharePoint web servers.

Client dialog boxes typically contain HTML markup, input controls, and JavaScript functions, and they interact with SharePoint through the JavaScript implementation of the client object model. This ASP.NET AJAX-based communication does not require page postbacks for every action that a user performs on data. Instead, commands are batched, and sent to the server for processing through **XmlHttp** operations. The client dialog box platform, therefore, provides responsive, efficient, Web 2.0–style dialog boxes that information workers use to interact with SharePoint data.

More information about developing ECMAScript (JavaScript, JScript) clients for the SharePoint Client Object Model:

[ECMAScript Class Library](#)

[Differences Between Managed and ECMAScript Object Models](#)

[Custom Action](#)

[Custom Action Definition Schema](#)

[Declarative Customization of the Server Ribbon](#)

[Default Custom Action Locations and IDs](#)

[Default Server Ribbon Customization Locations](#)

[How to: Modify the User Interface Using Custom Actions](#)

[Walkthrough: Replacing a Button on the Server Ribbon](#)

Client Authentication in Solutions That Use Remote APIs

When you are deploying server-side code as part of a sandboxed solution (such as in a Web Part or event receiver), you do not normally have to consider how users are authenticated by SharePoint Online. Users will already have logged on to your site collection by using their Windows Live IDs, and your server-side code automatically runs in the context of the logged-on user. However, if your solution uses the client object

model, then you may need to consider how authentication works in SharePoint Online, and how to interact with the authentication provider in your code.

Note: SharePoint Online supports only the Windows Live ID authentication provider.

Given that there are three different implementations of the client object model, and given that they can be used in various scenarios, you must understand when you might need to interact with the authentication provider in your code. The following scenarios outline these details.

Authentication in the ECMAScript Implementation of the Client Object Model

In this scenario, the client object model is executed by ECMAScript (JavaScript, JScript) that is embedded in the SharePoint page. Therefore, because the browser has already been authenticated, any code that you write to use the ECMAScript implementation of the client object model automatically uses the authentication cookie that is managed by the user's browser. Therefore, you normally do not need to consider how users are authenticated by SharePoint Online for the operations performed through the ECMAScript implementation of the client object model.

Authentication in the Silverlight Implementation of the Client Object Model

In this scenario, as long as the Silverlight XAP file is served from the SharePoint Online domain (for example, www.contoso.com) or site, the client object model will use the same authentication cookies from the browser session. Therefore, you normally do not have to consider how users are authenticated by SharePoint Online for the operations performed through the Silverlight implementation of the client object model.

Authentication in the .NET Implementation of the Client Object Model

In this scenario, your .NET code does not automatically get authenticated. Therefore, you need to provide a mechanism for the user to log in to SharePoint Online so that you can then use the authentication cookie with your **ClientContext** object.

First, be aware that the user must log on interactively, so you have to include a **WebBrowser** control in your .NET Framework application (such as in a Windows Forms or WPF user interface), and have the user employ that control to log on to SharePoint Online. When they are authenticated, the **WebBrowser** control will have received the authentication cookie supplied by SharePoint Online. However, these cookies are

marked as **HTTPOnly**, and therefore cannot be accessed directly by your .NET Framework code. Instead, it is necessary to make a call to the WININET.DLL. The .NET Framework can call COM-based DLL methods through **P/Invoke**, and the method you need to call is **InternetGetCookieEx**. This can return regular cookies *and* those with the **HTTPOnly** flag.

When you have retrieved the cookie, you can take the same approach as in Silverlight applications for adding it to the **ClientContext** object's request.

SharePoint Online Web Services

SharePoint Online web services are a subset of the SharePoint Foundation web services APIs. These APIs provide methods that enable you to access SharePoint Online data from Windows applications, custom applications, and other instances of SharePoint.

In SharePoint Online applications, you must use the Data Form Web Part to connect to any web service.

Note: Whenever possible, we recommend that you use the new client-side object model to work remotely with SharePoint Foundation data, instead of using the legacy ASP.NET web services.

Available SharePoint Online Web Services

The following table lists each of the web services that are available in SharePoint Online, its path, and a description for the web service.

Web service	Path from site	Description
Alerts	/_vti_bin/alerts.aspx	Provides methods for working with alerts for list items in a SharePoint Online site.
Copy	/_vti_bin/Copy.aspx	Provides methods for: <ul style="list-style-type: none"> • Copying items between locations in the SharePoint environment. • Adding files to a distribution list. • Copying files from one distribution list to another. • Downloading files from a distribution list.

Document Workspace	/_vti_bin/DWS.asmx	Provides methods for managing Document Workspace sites and the data they contain. Note: The FindDwsDoc method of the Document Workspace service is not available in SharePoint Online.
Imaging	/_vti_bin/Imaging.asmx	Provides methods that enable you to create and manage picture libraries.
Lists	/_vti_bin/Lists.asmx	Provides methods for working with lists and list data Note: The AddDiscussionBoardItem method of Lists web service is not available in SharePoint Online.
Meetings	/_vti_bin/Meetings.aspx	Provides methods that enable you to create and manage Meeting Workspace sites.
People	/_vti_bin/People.asmx	Provides methods for working with security groups.
Permissions	/_vti_bin/Permissions.aspx	Provides methods for working with the permissions for a site or list.
Site Data	/_vti_bin/SiteData.asmx	Provides methods that return metadata or list data from sites or lists.
Sites	/_vti_bin/sites.asmx	Provides methods for returning information about the site templates for a site collection.
Search	/_vti_bin/spsearch.asmx	Provides methods for remotely performing searches within a SharePoint Online deployment.
Users and Groups	/_vti_bin/UserGroup.aspx	Provides methods for working with users, site groups, and cross-site groups.
Versions	/_vti_bin/versions.aspx	Provides methods for working with file versions.
Views	/_vti_bin/Views.aspx	Provides methods for working with views of lists.

Web Parts Pages	/_vti_bin/webpartpages.aspx	<p>Provides methods for working with Web Parts.</p> <p>Note: The following methods of a Web Parts pages web service are not available on SharePoint Online:</p> <ul style="list-style-type: none"> • AssociateWorkflowMarkup • ExecuteProxyUpdates • GetAssemblyMetaData • GetDataFromDataSourceControl • GetFormCapabilityFromDataSourceControl • RemoveWorkflowAssociation • ValidateWorkflowMarkupAndCreateSupportObjects
Webs	/_vti_bin/Webs.aspx	<p>Provides methods for working with sites and subsites.</p> <p>Note: The CustomizeCss method of a Webs web service is not available in SharePoint Online.</p>
Publishing Service	/_vti_bin/PublishingService.aspx	<p>Provides methods to work remotely with the publishing service.</p> <p>Note: The following methods of the publishing service are not available on SharePoint Online:</p> <ul style="list-style-type: none"> • ExportObjects • GetObjectStatusCollection • GetObjectStatusCollectionWithExclusions • ImportObjects

More information about Web Services in SharePoint Online:

[SharePoint 2010 Web Services](#)

Conclusion

SharePoint Online provides a business collaboration platform on which you can build solutions to meet your business requirements. Depending on the functionality and degree of customizations you require, this Developer Guide shows how you can use a

variety of approaches. These approaches range from making browser-based customizations, to modifying and extending your site collections by using SharePoint Designer 2010, to deploying custom sandboxed solutions that you create by using Visual Studio 2010.

Appendix A – Setting Up Your Local Environment for SharePoint Online Solution Development

If you are developing sandboxed solutions that will be deployed to SharePoint Online, you will need to set up your local environment so that you can develop, test, and debug your solution before it is uploaded to the solution gallery in your SharePoint Online site collection.

You cannot deploy solutions directly from Visual Studio to SharePoint Online, and you cannot attach the Visual Studio 2010 debugger to solutions that are deployed in SharePoint Online, so you must attempt to mirror the SharePoint Online environment as closely as possible in your local environment.

If you are new to SharePoint development the easiest way to get set up with all the tools and products you need is to use the SharePoint 2010 Easy Setup Script. The SharePoint 2010 Easy Setup Script is a new set of prepackaged tools that help developers get started with SharePoint 2010 development quickly by automating the provisioning of a developer workstation using Windows 7, SharePoint, and associated tools:

[SharePoint 2010 Easy Setup Script](#)

If your development computer is Windows Server 2008, you can install a SharePoint Foundation 2010 server by using the following guide:

[Deploy a single server with a built-in database \(SharePoint Foundation 2010\)](#)

However, SharePoint Foundation 2010 is also supported for developers on 64-bit versions of Windows Vista and Windows 7. If you want to use Windows Vista or Windows 7 for your development of SharePoint Online solutions, you can install SharePoint on your development computer by using the following guide:

[Setting Up the Development Environment for SharePoint 2010 on Windows Vista, Windows 7, and Windows Server 2008](#)

After you have installed SharePoint Foundation on your development computer, you must ensure that the Sandboxed Code Service is running, as follows:

1. On the **Start** menu, click **All Programs**, click **Microsoft SharePoint 2010 Products**, and then click **SharePoint 2010 Central Administration**.
2. In the **System Settings** section, click **Manage services on server**.
3. In the list, next to the **Microsoft SharePoint Foundation Sandboxed Code Service** item, verify that the service is started, or click **Start** to start the service.

After you have completed these steps, you can develop, test, and debug sandboxed solutions easily in your local environment. Then, when your solutions are stable, you can package them into .wsp files and upload and activate them in the solution gallery in your SharePoint Online site collections.

However, it is recommended that before you deploy your solutions to a production site collection in SharePoint Online you should perform final verification of their usability in a site collection in SharePoint Online that mirrors your production site collections, but which is not used by your information workers. This approach will enable you to perform final tests and verifications for your solutions before you deploy them to the production site collections.