PART I

Getting Started with SharePoint 2013

COPYRIGHTED MATERIAL

1

Introduction to SharePoint 2013

WHAT YOU WILL LEARN IN THIS CHAPTER:

- Understanding what SharePoint is
- Learning about the high-level feature areas and functionality of SharePoint 2013
- Understanding the relevance of these feature areas and functionality to the developer

SharePoint 2010 launched a major evolution in the product's life — it was a first-class platform that enabled you to not only leverage a wide array of out-of-the-box features to manage collaboration, but it also provided a rich development platform. This made developing solutions powerful and relatively straightforward. With SharePoint 2013 arrives a new paradigm shift, one much more closely aligned to Microsoft's overall shift to the cloud. For those of you who have been on the SharePoint train for some time, this means thinking in a slightly different way about how you develop applications for SharePoint. For those who are new to SharePoint, welcome. You're in for one heck of a ride!

SharePoint is an exciting Web-based technology. In its fifth version, SharePoint has undergone quite a transformation from the initial releases, and the types of things you can do with SharePoint run far and wide. Those who have had the chance to see the product grow up will be surprised and happy with many of the changes that are now built into the platform. In fact, existing SharePoint developers will witness what arguably is a sea-of-change in the features and functionality that SharePoint provides, as well as an evolution in the tools supported and the developer community that rallies around the technology. Aspiring SharePoint developers will realize quite a bit of power exists in the platform and should have the capability to put it into practice by the end of this book.

SharePoint is maturing into a *cloud-centric* platform that will enable you to build and deploy a wide array of solutions, as well as take advantage of the build-and-publish model that SharePoint users and developers have come to enjoy. It has also evolved into a platform that is much more open by design. This means that developers are moving beyond what was predominantly an ASP.NET- or JavaScript-based development approach. In SharePoint 2013, you have the ability to bring your own hosted Web applications and technologies to the table and use OAuth authentication and registration hooks that are built into SharePoint to integrate those apps into the SharePoint experience. This is a significant evolution, and one not to be undersold.

Similar to SharePoint 2010, SharePoint 2013 offers such a wide array of features that claiming to be an expert across all the workloads will be challenging for any one person. You will need to dedicate some time to become an expert, but the journey will be worth it.

With that in mind, this chapter introduces you to what SharePoint is and walks through some of the high-level areas for the developer. This chapter also answers the question of what capabilities make SharePoint a platform that is interesting and compelling for you, the developer, to learn. It also helps you understand why SharePoint 2013 is evolving to the cloud.

Specific topics include discussion around programmability, new app models, platform services, and the ways in which you can build and deploy a SharePoint solution.

GETTING TO KNOW SHAREPOINT

Simply put, SharePoint 2013 (also referred to as SharePoint after this point) is a platform to support collaboration — a central Web-based portal for you to manage your own and your colleague's documents, social activities, data, and information. This definition is pretty broad, but try framing it within a scenario: you manage projects on a daily basis and must also manage teams of people across those projects. Within the project, people are having meetings, creating documents, exchanging ideas, managing schedules, and so on. Without a central place to manage these activities and documents, you're using file shares on servers; you're exchanging documents via mail; and you're using one or more different types of management software to help keep a common view of activities. Within this one scenario, you should be able to see the problem. A file share can go down anytime, so what's the backup? Documents aren't versioned. Context is lost around a project as elements are spread out across different technologies. And security around those documents is difficult to manage and control in an effective in an effective and efficient way.

Project management is but one scenario that paints a picture of collaboration. Many others exist, and this is why SharePoint has seen such broad adoption. Often companies see great advantages with SharePoint through simple document management; that is, being able to store, version, create, and manage documents in one central place. However, what these companies soon discover is that many more features are built into SharePoint such that its use goes beyond simple document management. Users soon begin to see Business Intelligence (BI) features, discoverability benefits (that is, search functions), social features, and governance abilities, among the many other areas of which they can take advantage.

Defining SharePoint by Function

To provide you with an idea of the types of things that you can do with SharePoint, Figure 1-1 breaks SharePoint out into three separate areas:

- **Collaboration —** As you read through this book, you'll see the notion of *collaboration* as a very strong theme for SharePoint. This is because SharePoint is about bringing people together through different types of collaboration, such as enterprise content management (ECM), Web content management (WCM), social-computing through the use of newsfeeds, discoverability of people and their skills, creating dashboards to fulfill your BI needs, and so on. Given the new app model in SharePoint 2013, collaboration is managed through apps. Developers can extend, customize, or build their own Apps for SharePoint as well manage collaboration on SharePoint.
- **Interoperability —** SharePoint is also about bringing this collaboration together through *interoperability*. This means Office and Web-based document integration, and the capability to build and deploy secure and custom solutions that integrate line-of-business (LOB) data with SharePoint and Office, integrating with wider Web technologies, or deploying applications to the cloud.
- **Platform —** As you'll see, SharePoint is a *platform* that supports not only interoperability and collaboration but also extensibility, through a rich object model, a solid set of developer tools, and a growing developer community. One of the key paradigm shifts here, though, is the notion of the cloud in SharePoint. The cloud introduces new app models: new ways of developing, deploying, and hosting SharePoint applications; new forms of authentication through OAuth; and new ways of data interoperability using OData (and REST).

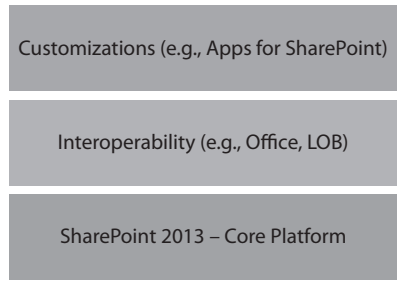

FIGURE 1-1

These are three key themes that you will find crop up throughout most discussions of SharePoint and implicitly through many of the capabilities you'll get to explore throughout this book.

So, at its essence, SharePoint is a Web-based platform that provides the following:

- A set of native, out-of-the-box capabilities to support productivity and collaboration
- An open and extensible set of APIs and services that you can use to build light apps or cloud-based apps using your own hosting technology
- Infrastructure to manage security and permissions against the various artifacts (for example, documents and list items)
- A management and configuration engine that provides deep administrative abilities, both for the cloud-hosted version of SharePoint and the on-premises SharePoint server.

Defining SharePoint by User

Depending on the role of the person who is using SharePoint, the stated definition might take on a slightly different hue.

For example, for the end user, SharePoint enhances productivity by providing a core set of connected applications that essentially act as the Web-based application platform. The applications enable people to connect using wiki sites, workspaces, lists, document libraries, and integration with Microsoft Office applications such as Outlook, Excel, and Word 2010.

From an organizational point of view, the unified infrastructure enables the organization to rally around a central point of collaboration — be it through an organizational portal, a team site, or a personal My Site. It also enables organizations to integrate LOB systems, such as SAP, Siebel, PeopleSoft, and Microsoft Dynamics, into the information worker experience through SharePoint. Furthermore, it enables you to tap into your growing cloud services and data that you might be developing and deploying.

From a developer's perspective, you can take advantage of a wide platform (arguably the widest historically for the platform) to build and deploy many different types of applications. These range from simple HTML and JavaScript applications to managed code and .NET cloud apps that are deployed to Windows Azure.

The response to business needs arrives through the capability to use SharePoint as a toolset in the everyday work lives of an organization's employees — for example, routing documents through managed processes, providing social newsfeeds and updates, or managing and tracking project documents. In essence, SharePoint represents a platform that offers the organization a lot of functionality to do many different things, with collaboration lying at the heart of them.

Introducing the User Interface

Taking a look at the SharePoint user interface at this point might be helpful for you. Although you can create sites from many different templates, Figure 1-2 shows a Team Site and calls out some of the areas of the page:

- Area 1 is where you can access other areas of Office 365 such as Outlook or the Site Settings.
- Area 2 provides a search box for you to enter queries and search the site collection.
- Area 3 contains some quick launch tiles that can help you get started with your site (note you can click the Remove This link to hide them).
- Area 4 provides a place for you to upload and view documents.
- Area 5 shows quick links to other areas of your Team Site.

You'll find a common set of options in many sites (such as the link bar at the top of the site). Depending on the type of site that you create, you'll find a different set of default options available. For example, some have more BI functions or governance workflow or social features built into them. This all depends on the type of site.

If you're a developer reading this book, you might be happy to know that many parts of the SharePoint development experience are customizable. For example, in Figure 1-2 you could programmatically add data from external LOB systems into your site, you could integrate a Web app from Windows Azure, or you could create a light HTML and JavaScript and deploy to your Team

site. You could also customize the branding of the site. For example, Figure 1-3 shows a sample SharePoint site that has more branding. This example uses some of the native SharePoint capabilities to configure the look and feel, but you could create a much more elaborate, branded, and custom look-and-feel for any of your SharePoint sites.

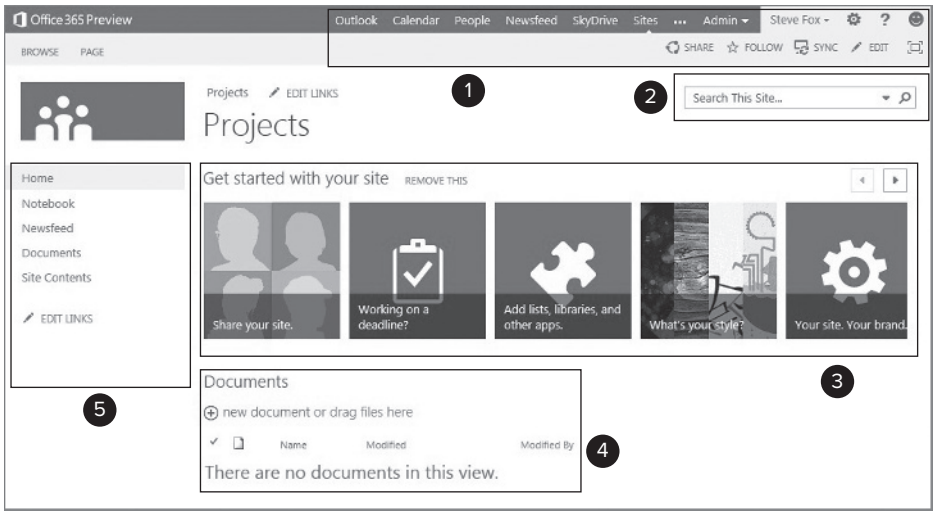

FIGURE 1-2

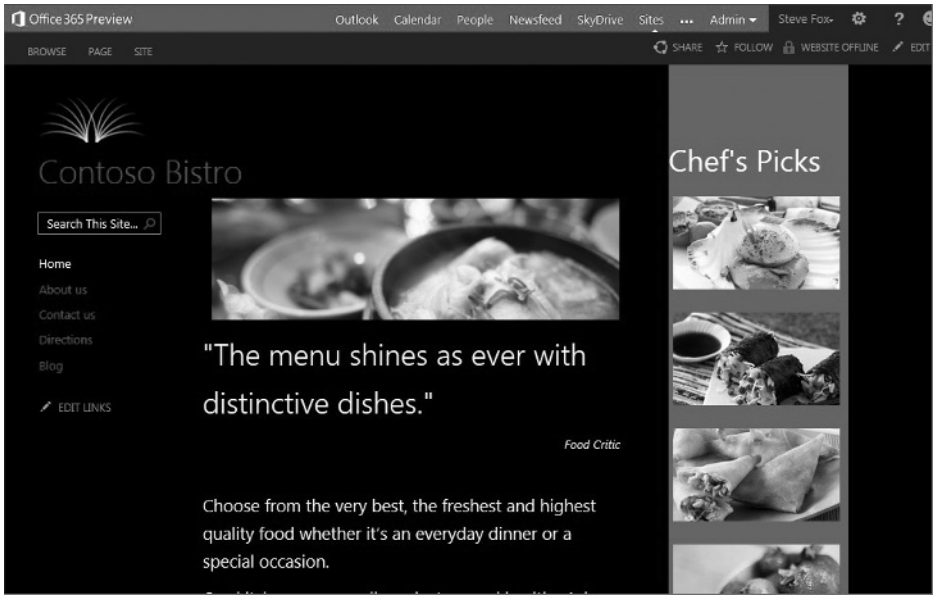

FIGURE 1-3

Thus, the Web-based experience that SharePoint provides out-of-the-box integrates core (as well as external) applications and functionality that end users can employ during their daily work lives.

In Figure 1-4, note that the default view has changed. This is because the site is now in Edit mode, which enables you to customize the SharePoint site. In this view, you can add Web parts, HTML or JavaScript apps, integrate external applications, and so on. The fact that you can quickly put a site into Edit mode, make some changes, and then save those changes back to the server is one of the great advantages of SharePoint.

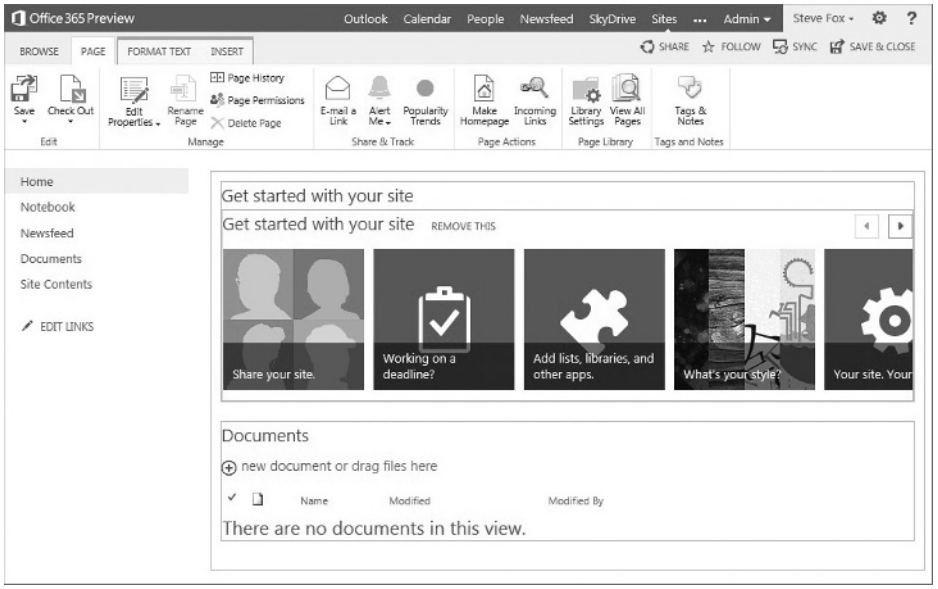

FIGURE 1-4

Introducing the Structure

The structural taxonomy of SharePoint comprises multiple levels. On the first level you have a site that is made of a template. As mentioned earlier, you have a variety of templates that you can use for a given site — either out of the box or custom. Within a site, you can create more subsites — using the same set of site templates. So it's essentially a parent site, or site collection, with subsites. Within a specific site, you then add (or create and deploy to the sites) *apps*. Now for those of you who have been around SharePoint for a while, this will feel a little weird: *Everything is now an app*. That is, lists, document libraries, form libraries, and so on are all apps — just different types of apps. For those who are new to SharePoint, this idea won't seem so jarring; thinking about a site comprising apps is a pretty natural way to think about Web platforms today. Also, as you start building apps for a marketplace, then the concept of an app (as opposed to differentiating across lists, document libraries, and so on) begins to make even more sense. Figure 1-5 shows you a small set of apps that are available to you by default within your SharePoint site.

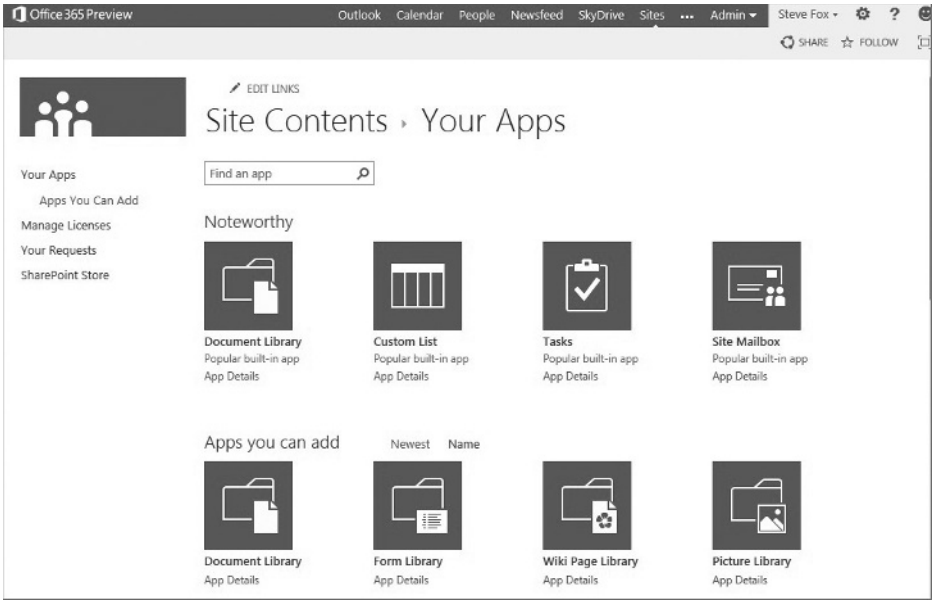

FIGURE 1-5

For organizations, SharePoint provides a one-stop shop for leveraging the SharePoint infrastructure not only for internal sites to manage your day-to-day project needs and as a business process workflow, but also activities and infrastructure to manage your publicly facing sites. The key point is that SharePoint provides the infrastructure for many types of sites and for site and app development.

As you'll see throughout this book, the native SharePoint experience is, in many ways, customizable. Given the breadth of integration possibilities with SharePoint 2013, there's an adjunct set of technologies including Windows Azure, PHP, and other Web technologies that might factor into your SharePoint development experience.

ADDRESSING THE NEEDS OF THE DEVELOPER

At its essence, SharePoint is a platform. And to see how SharePoint can help you as a developer, you must understand those platform capabilities. When you explore and learn the range of functionality that make up the platform, you'll begin to see some interesting and compelling opportunities emerge for the developer.

Take a look at a practical example. As you have seen, a business productivity platform implies having a platform for end users to make them more collaborative and productive in their day-to-day work lives — and SharePoint can certainly do that. In short order, it can be used as an application for end users. For example, a Human Resources (HR) department might use SharePoint to manage employee reviews, or a sales team might use it to manage a monthly sales-forecasting dashboard for BI.

In all of these scenarios, SharePoint first represents an end user collaboration platform, and second represents a base that skilled developers can augment or extend. So, when your sales manager comes

to you (the developer) and asks you to design a SharePoint site collection that integrates daily sales data from an SAP system and plot high-potential markets on a map in the SharePoint site — so salespeople can see current sales pipeline versus opportunity areas — you wonder in what ways this type of app would manifest in SharePoint.

Let's look at this task from two perspectives:

- **End users** want a site they can open, manage their sales documents and spreadsheets, filter and pivot data in, and then get a quick view on the map to see where they should be targeting. They want ease of use and actionable apps.
- **Developers** want to craft an experience that is easy to use and intuitive — but also efficient to manage.

As a developer, you'll want to keep both perspectives in mind when performing the task. In doing so, implementing a solution for each task will likely require leveraging a combination of native features — such as document libraries and lists — and core services, capabilities, and APIs built into the platform to get you to the next level from a development perspective. You can also integrate either third-party or your own custom cloud-based services to round out the development experience.

For this particular example with your sales manager, you could use a combination of Business Connectivity Services (BCS), which is a set of services within SharePoint that enables you to connect to LOB systems and Excel Services, so you can create "pivotable" spreadsheets for salespeople. The end result of using BCS is a dynamically generated list app to contain the sales data and a document library app where you would house the spreadsheets. Therefore, you need to create two types of apps — a document library that leverages Excel Services and an external list app that loads the external LOB data. You could then integrate an HTML or JavaScript-based app that uses a cloud-deployed service to create a Bing map, and then overlay pushpins that are color-coded green for high potential, and red for low potential (or saturated) markets. This app would be the third one needed — but behind it sits a service you're plugging into (such as the Bing Maps service) and your own custom service that has the logic to create the pushpins based on some set of business rules or information. You can accomplish the development and deployment of these three apps either using apps that are deployed to an existing Team site, or by creating your own custom site with the Sales department branding.

The key takeaway from this example is that depending on what your audience requires, you can use SharePoint to create interesting experiences. You should be thinking about all these options as you design and build your SharePoint experience.

Extending SharePoint 2013

Although SharePoint represents a set of connected apps and functionality, it still has a vast array of opportunities for developers to extend and enrich the end-user experience at multiple levels. This experience is obviously important when you think about SharePoint in the context of the enterprise developer. However, when independent software vendors (ISVs) think about the custom experience they want to deploy to their customers, having a reliable platform beneath their feet that they can

deploy to and use to customize their SharePoint solutions becomes vital. Furthermore, they require a place to monetize; that is, a marketplace that provides not only a place for deployment and advertising, but also a place for in-product or catalog integration. Their business depends on platform stability, predictability, accessibility, and discoverability. So what does it mean to extend SharePoint 2013?

With the entry and integration of broader cloud-hosted models, extending and building on SharePoint means a wider array of Web development partners, customers, and ISVs can participate in the SharePoint phenomenon. Some of these forms of participation include the following:

- Building "light" apps (for example, HTML and JavaScript apps) for SharePoint
- Leveraging the new cloud-hosted app models to either build Windows Azure–based apps or use technologies from a broader set of Web standards and technologies

To further understand this extensibility in a paper available through Forester Research (`www.forrester.com/rb/Research/now_is_time_to_determine_sharepoints_place/q/id/45560/t/2`) entitled, "Now Is the Time to Determine SharePoint's Place in Your Application Development Strategy," John R. Rymer and Rob Koplowitz reinforce a model of SharePoint 2013 that is composed of different layers. The two authors propose that SharePoint has an *application layer*, where end users integrate with the out-of-the-box collaboration and productivity applications; a *customization layer*, where either power users or developers can begin to customize the SharePoint experience for the end user; and a third layer, which is the *application development layer*.

This application development layer is where things get very interesting for developers. At this layer you'll mostly find the solution developer who builds and deploys (or integrates through existing SharePoint artifacts such as Web parts or event receivers) applications or business solutions. What's also interesting is how this application development layer has evolved. Figure 1-6 illustrates how SharePoint 2013 has evolved from earlier application development paradigms.

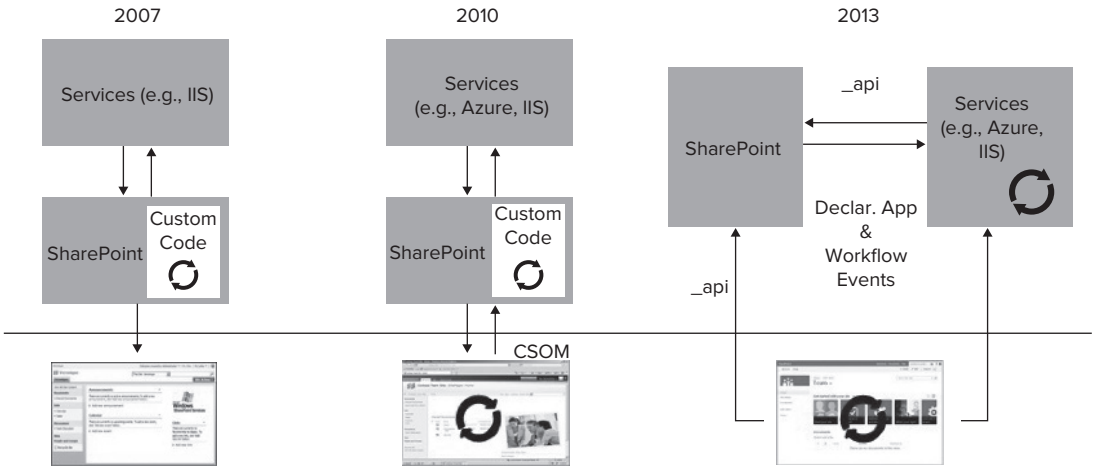

FIGURE 1-6

Figure 1-6 shows that SharePoint 2007 ran custom code or services from IIS or other servers. The custom code for the most part ran within an application pool using IIS resources. In 2010, SharePoint supported running on IIS (or other servers) and also introduced sandboxed solutions and the Client-side Object Model (CSOM), which then enabled contained solutions and client-side code to run. This version also brought the introduction of a Windows Azure that was more integrated with the SharePoint development paradigm — both on the server and on the client. In 2013, this development paradigm takes Windows Azure integration to the next level — in some cases natively using Windows Azure as the deployment, storage, and computing mechanism. Figure 1-6 illustrates workflow running in Windows Azure and hooking into SharePoint through a refactored REST API (`_api`). Thus, at the application development layer quite an evolution has occurred within SharePoint 2013.

> ***NOTE*** *If you're not familiar with Windows Azure yet, don't worry. Read Chapter 5, "Overview of Windows Azure for SharePoint"; you'll also see lots of examples throughout the book that introduce you to this new cloud technology from Microsoft.*

In light of these different layers, extending SharePoint means something slightly different in SharePoint 2013 than in past versions, including:

- A more open approach to development
- A broader integration with the cloud
- Support for open source and non-Microsoft technologies
- Bringing your own hosted apps to the SharePoint experience (Think of the Facebook app model: Facebook is a rich social platform that enables you to run apps, but those apps don't run within Facebook; they just consume parts of Facebook.)

Breaking It Down for Developers

As you might have gathered by now, SharePoint development can mean a number of things. For example, if you want to simply add an app to a page, you might consider yourself a developer. If you customize the branding of a SharePoint site, you might only have to interact with page layouts or master pages (that is, the way in which you structure content in SharePoint) but you still may be a developer. Finally, if you do deeper-level solution development, you might be creating HTML5 and JavaScript applications that interact with SharePoint through native APIs, or use .NET and the cloud-hosted app model with Windows Azure. This type of development would mean you're a developer who uses Visual Basic or C# along with potentially leveraging different application programming methods such as Model, View, and Controller (MVC) apps and REST services. As you delve more into the managed-code side of the house to build your SharePoint apps, you will, of course, enter into a more complex development paradigm.

> ***NOTE*** *This book doesn't get into a lot of non-Microsoft Web technologies that you could use with SharePoint. However, you can use many different types of open-source, third-party, or non-Microsoft technologies to build Web applications that you could then integrate back with SharePoint.*

Exploring the Different Levels of SharePoint Development

The point is that there are different levels of "development" in regards to SharePoint, and each level serves the *end user* of the SharePoint site in some way. One way of looking at it is to think of development as cutting across a spectrum with the following areas:

- **Power user:** Someone who has advanced privileges on a SharePoint site, administers permissions, manages administration of a SharePoint site, manages apps on the site, and might even create lightly customized sites for consumption.
- **Designer:** Someone who is largely in charge of branding and master page customizations, designing the user experience, designs graphics for the site, implements CSS or other style sheets, and so on.
- **Website developer:** Someone who develops managed code, mark-up code, or unmanaged/client-side code solutions for SharePoint sites. This is you!

Although those of you who have a SharePoint background might split this spectrum even further, development in general can fall within the preceding three areas. You might argue that the people performing both tasks of site-branding and app development are equally identified as developers on the SharePoint platform, but the fact is that actual development can range from using HTML5 and JavaScript to .NET and service-based technologies (that is, REST or WCF) to non-Microsoft Web technologies. This spectrum is not only symptomatic of SharePoint being a broad platform but also a symptom of the different standards, applications, and interoperability that SharePoint must support as a good citizen of the Web. Web interoperability is even more important with the 2013 release given the focus on cloud-hosted apps.

If you break down these levels of development and use across Rymer and Rob Koplowitz's different layers of SharePoint, you'll find that the largest population of SharePoint consumers interacts with the applications layer. These consumers are the end users, and they represent your core audience for building and deploying your custom applications to SharePoint. Power users of SharePoint might operate at the customization layer because they possess a high degree of SharePoint knowledge.

Then there is the Web (or SharePoint) developer. You are, in many cases, the person who develops those custom applications for SharePoint or the next killer app in the ISV ecosystem. You are also the one for whom this book was written. In some cases, you as the developer might collaborate with the power users or designers, and in others you will work independently of one another.

As a power user, designer, or Web developer, you have a number of development tools at your disposal. They range from in-browser tools, for example, Napa, to designer tools such as SharePoint Designer, to more traditional development IDEs, including Visual Studio 2012, that support managed, unmanaged, and client-side code; debugging; ALM; and so on.

With regard to developer productivity, this means that you can use either Visual Studio 2012 or SharePoint Designer (SPD) as your core set of developer tools. As a professional Web developer, you'll likely use Visual Studio as your core toolset — especially if you're a .NET programmer looking to get into the SharePoint space. As for SPD, you're more than likely going to use it to edit master pages and page layouts, as well as to build noncomplex workflows using a Visual Rules approach (for example, using Visio 2013 and SPD). As a complement to these tools, you might also use Expression Blend either as a way to build more advanced and interactive UIs (through Expression Blend) or through Expression Web for baseline Websites.

> ***NOTE*** *Chapter 3, "Developer Tooling for SharePoint 2013," explores developer tools in more detail.*

In terms of rich platform services, SharePoint 2013 offers the developer a wide array of methods for getting, managing, and updating objects and data within a SharePoint site. With this version of SharePoint you'll see increased investments in REST and OData, app authentication through OAuth, and, of course, a host of client-side APIs using the client-side object model to enable many different types of application programming and solution development. In this book, you'll discover new application programming interfaces (APIs), new investments in the developer, and new services that will enable you to build many different types of apps, and you'll also learn about how to enable LOB system integration to bring external data into your SharePoint applications.

> ***NOTE*** *This book is divided into three parts, each of which covers these new areas in increasingly greater detail and at increasingly advanced levels.*

Deploying Your Application

After you build your application, you need to deploy it. In SharePoint 2013, you can deploy two primary types of apps: Apps for SharePoint and SharePoint Solutions.

Those who have been around SharePoint before might recognize Solutions: they are the Windows SharePoint Services Solution Packages (WSPs) that represent small- to large-scale packages that are used to customize or augment SharePoint sites in some way. SharePoint Solutions are typically run as full-trust solutions and require a farm-level deployment. In SharePoint 2010, you could also run Solutions (.WSP) in a partial-trust sandboxed environment, and although this sandboxed environment still exists in 2013, it will be deprecated in the future.

> ***NOTE*** *Because SharePoint deployments typically comprise Web front-end servers, application servers, and database servers, the deployment and configuration of these servers is called a* SharePoint farm. *WSPs run at the farm level, meaning you can install and manage them across the entire SharePoint farm. Sandboxed solutions run in a special partial-trust environment that has its own measures and quota to ensure the application is isolated.*

Apps for SharePoint (.APP) are new to SharePoint 2013 and are standalone applications that provide specific configuration information and functional components to a SharePoint site. Apps for SharePoint are easy to install, use, manage, upgrade, and delete. You can add Apps for SharePoint from a corporate catalog or the Marketplace. You can also leverage two different hosting models: one that is a lighter app and is hosted within SharePoint (think HTML and JavaScript apps) and one that is hosted within the cloud-hosted model (think Windows Azure–hosted apps).

Within these different types of SharePoint deployment techniques, you can do the following:

- Import a standard Windows SharePoint Services Solution Package (WSP) into your SharePoint farm.
- Build and deploy a solution to a SharePoint instance within the corporate firewall.
- Build and deploy solutions to a SharePoint site hosted on the wider Internet.
- Package and deploy the .APP to the cloud, but configure and register it to load in SharePoint.

As you think about SharePoint 2013 development, keep the following things in mind:

- SharePoint's new direction is more cloud-centric. You should be thinking about this from design to deployment.
- SharePoint has a rich object model, as well as a set of services and APIs that you can leverage when developing custom solutions.
- Visual Studio 2012 has a mature, out-of-the-box experience for building and deploying SharePoint solutions.
- You can build and debug SharePoint sites remotely.
- A number of ways are available to interact with SharePoint data (for example, the client-side object model).
- You can leverage BCS to build rich LOB apps.
- Multiple integration points exist across other Microsoft and third-party applications (such as Office 2010, SAP, PeopleSoft, Microsoft Dynamics, Microsoft Silverlight, and so on).
- A cloud-based deployment methodology now exists for SharePoint 2013 that is defined using the .APP deployment.
- You can deploy SharePoint 2010 solutions on premises or to the cloud (that is, SharePoint Online). However, the future is deployment to the cloud.

These points represent just a sampling of what you can do with SharePoint, and the goal of this book is to show you how you can get started with all of these tasks and more. Keep in mind that when SharePoint references business productivity, it not only means for the applications that you'll be building and customizing for your end users, but also for the developers as you build apps that deploy into this platform for business productivity.

SHAREPOINT 2013: THE PLATFORM

SharePoint maintains a high-level architecture that is made up of a number of components (see Figure 1-7). You first install the core software on Windows so you can create SharePoint farms. A SharePoint farm is, in essence, one or more servers that make up your SharePoint instance. As a developer you should understand the three-tiered structure and roles of the SharePoint farm architecture, which includes a Web server role (a fast, load-balanced, lightweight server that responds to

user requests and loads Web pages), Application server role (which provides the service features for SharePoint such as Excel Services), and Database server role (which stores content and service data). Your apps may interact with any one or all of these server roles.

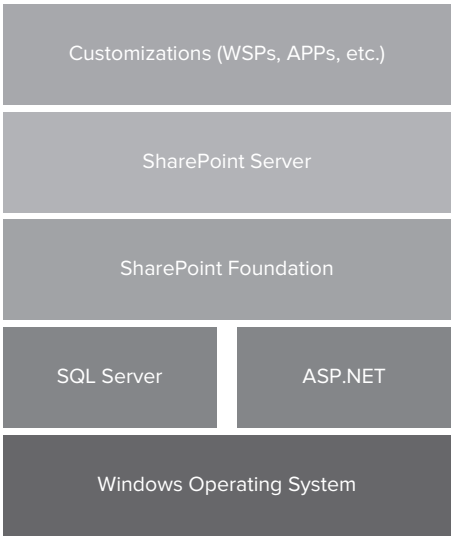

FIGURE 1-7

You can have a standalone server acting as the entire farm (for example, all the components listed in Figure 1-7 installed or working on one machine). For testing and light workloads, this configuration might be adequate, depending on the hardware specifications. For larger organizational deployments, inclusive of failover and redundancy, a one-server farm is not adequate. However, the Windows operating system is your underlying install base — specifically, Windows Server 2008, 2008 R2, and Windows Server 2012. SharePoint heavily leverages SQL Server as its underlying content database and ASP.NET/IIS as the application service server. You can then install either SharePoint Foundation (the free version) or SharePoint Server (which is loaded with enterprise-grade features), on top of which you would build and install your customizations. Or, as an alternative to installing SharePoint Foundation or SharePoint Server, you can sign up for Office 365, which provisions and manages the underlying infrastructure for you but still gives you the power of programmability.

SharePoint Installation Types

When you install SharePoint, you can choose different types of deployments and installation types. There are three main ways to install and use SharePoint.

SharePoint Foundation

SharePoint Foundation ships as a free, downloadable install and represents the foundational parts of SharePoint. It includes a number of features such as security and administration, user and Team site collaboration, and a number of apps (such as document libraries and lists). In essence, it provides a baseline set of features that enable you to get started with both using and developing for SharePoint.

Although the functionality that ships in SharePoint Foundation is less broad than that which ships in SharePoint Server, downloading and installing SharePoint Foundation costs you nothing. You can get up and running very quickly with this version and begin your development work using it. In SharePoint 2013, though, you also have the ability to create SharePoint Online sites very quickly — and have a rich development model there as well.

SharePoint Server

SharePoint Server offers a wealth of features that extend upon those offered in SharePoint Foundation. These features include additional app types, Office server-side services such as Word and Excel Services, enhanced search versions, enhanced BI, and much more.

> ***NOTE*** *You can get more information from an IT pro perspective on topics such as what's new in SharePoint 2013, installation methods, farm architecture, and more from the following TechNet article:* `http://technet.microsoft.com/en-us/sharepoint/fp142366.aspx`.

The following list provides a sampling of some of the services available in SharePoint Server:

- **Access Services:** Allows creation of new Access service applications using the Access 2013 Preview client. View, edit, and interact with Access Services databases in a browser.
- **Access Services 2010:** Allows continued maintenance of SharePoint 2010 Access service applications by using Access 2010 clients and Access 2013 Preview clients. Does not allow users to create new applications.
- **App Management Service:** Allows you to install apps from the internal app catalog or the public SharePoint store.
- **Business Data Connectivity:** Access line-of-business data systems.
- **Excel Services:** View and interact with Excel files in a browser.
- **Machine Translation Service:** Performs automated machine translation.
- **Managed Metadata Service:** Access managed taxonomy hierarchies, keywords, and social tagging infrastructure as well as content type publishing across site collections.
- **PerformancePoint:** Provides the capabilities of PerformancePoint Services.
- **PowerPoint Conversion:** Converts PowerPoint presentations to various formats.
- **Search:** Crawls and indexes content and serves search queries.
- **Secure Store Service:** Provides single sign-on authentication to access multiple applications or services.
- **State Service:** Provides temporary storage of user session data for SharePoint Server components.
- **Usage and Health Data Collection:** Collects farm-wide usage and health data and provides the ability to view various usage and health reports.
- **User Profile:** Adds support for My Sites, profile pages, social tagging, and other social computing features.
- **Visio Graphics Service:** Views and refreshes published Microsoft Visio diagrams in a Web browser.
- **Word Automation Services:** Performs automated bulk document conversions.
- **Work Management:** Provides task aggregation across work management systems, including Microsoft SharePoint Products, Microsoft Exchange Server, and Microsoft Project Server.
- **Microsoft SharePoint Foundation Subscription Settings Service:** Tracks subscription IDs and settings for services that are deployed in partitioned mode. Windows PowerShell only.

You can also choose to purchase the Internet-specific edition, SharePoint for Internet Sites, which provides rich publishing templates and workflow that you can use to create and deploy SharePoint sites to the wider Web (for example, building a scalable SharePoint site for public, anonymous access).

Office 365

Office 365 has emerged as a third, fully cloud-hosted model for SharePoint — as opposed to hosting your own farm in your own on-premises Data Center. It has also become a great place where you can develop rich applications (both as SharePoint-hosted and cloud-hosted apps) and scale without the cost of managing the on-premises infrastructure. It doesn't have all the same services and features as SharePoint Server, but does carry with it some great development capabilities.

As a developer, you have the capability to customize any of the SharePoint editions, whether it's SharePoint Foundation, Server, or Office 365. For example, beyond thematic or branding customizations, you can also develop and deploy custom solutions to each of these SharePoint versions. There are .NET applications that you build using C# or Visual Basic and then deploy into SharePoint as .WSPs or .APPs, or there are lighter-weight apps such as HTML5 and JavaScript apps that you can also deploy. What's important to understand is how that customization opportunity varies across the different versions; you'll explore this throughout the book to understand how to choose across these options.

SharePoint 2013 Capabilities

A default set of capabilities (or features) is built into SharePoint that enables you to take advantage of the platform without doing any development. You can also use or extend these core capabilities when building your apps. Microsoft has historically referred to these capabilities as *workloads*. These workloads provide a way to talk about the different capabilities of SharePoint coming together, and you should see these workloads as not only representing a core set of related applications but also as opportunities for your application development.

For those who are experienced SharePoint developers, you'll remember that Microsoft described the core capabilities for the SharePoint through workloads (seen in many 100-level presentations on SharePoint). In SharePoint 2010, these workloads were:

- **Sites:** Representing the different types of sites available for use and the features within these sites
- **Communities:** Representing the community and social features such as blogs and wikis
- **Content:** Representing core enterprise content management features
- **Search:** Representing the search-driven features
- **Insights:** Representing business intelligence features such as KPIs
- **Composites:** Representing the ability to integrate external applications by using, for example, Business Connectivity Services

These previous workloads have not gone away in SharePoint 2013; moreover, Microsoft has extended them to add more features and provide tighter integration.

Table 1-1 lists a sampling of the core capabilities for SharePoint 2013. Those of you who are experienced developers will see a lot of familiar areas because a lot of what you had in SharePoint 2010 is still available in SharePoint 2013, with a number of added areas. For example, note from the services listed previously in the "SharePoint Installation Types" section that Machine Translation Service, Access Services, App Management Service, and Work Management Service are new to SharePoint 2013. Furthermore, rather than Office Web Apps being a service, it is now a separate server product — which for IT pros will impact the design of your SharePoint farm topology. Also, what was FAST search in 2010 as a separate server product has been subsumed within SharePoint 2013 — which is fantastic because it improves the search experience immensely in this release. The whole movement to the cloud in general is a major shift in the way of thinking about SharePoint development; it is simultaneously exciting and challenging as developers need to think about app design and deployment in different ways than before.

Each of the example capabilities in Table 1-1 offers many different development opportunities.

TABLE 1-1: Sample SharePoint Capabilities

CAPABILITY	NATIVE FEATURES	EXAMPLE EXTENSIBILITY
Sites	Sites is where you'll predominantly find the collaborative aspects of SharePoint. Sites contain an abundance of features, including the capability to create, store, and retrieve data, and manage, tag, and search for content, documents, and information. You also have connectivity into the Microsoft Office 2013 client applications through the list and document library.	Sites, site templates, Apps for SharePoint, workflow, master pages, site pages
Social	Provides social and social networking capabilities, newsfeeds, and profile searching and tagging, along with the capability to search, locate, and interact with people through their skills, organizational location, relationships, and rating of content.	Search customization, rating and tagging capabilities, blogs, wikis, metadata tags
Content	Contains the capability to explore, search, and manage content using Web pages, apps, workflow, or content types.	Apps for SharePoint, workflows, Word or Excel Services
Search	The ability to search content inside and outside of SharePoint in a rich and dynamic way with real-time document views through Office Web Apps. Also, the integration of information in structured database systems and on-premises or cloud-based LOB systems such as SAP, Siebel, and Microsoft Dynamics.	SharePoint Search, Search customization, Business Data Connectivity (BDC)

continues

TABLE 1-1 *(continued)*

CAPABILITY	NATIVE FEATURES	EXAMPLE EXTENSIBILITY
Insights	Predominantly about BI and support, for example, the capability to integrate Microsoft Access into SharePoint; leverage Excel and SQL Server to access and display data on a Web page; enable the use of dashboards and key performance indicators (KPIs) to transform raw data into actionable information.	Excel Services, Access Services, dashboards, BDC, PerformancePoint Services
Interoperability	Ranges from LOB integration to Office integration through the new Apps for Office application model (think HTML and JavaScript-fueled custom task panes that link to cloud services instead of VSTO managed code add-ins) to custom solution development.	BDC, Apps for Office, custom development
Branding	Changing the look and feel of your site through built-in template changes or more detailed and organizationally driven branding.	Out-of-the-box configuration (for look and feel), master pages and customized Apps for SharePoint

You will discover many more ways to develop for SharePoint as your journey deepens and you become more familiar with all the different facets of the SharePoint capabilities. For a complete list of updates for SharePoint 2013, visit: `http://technet.microsoft.com/en-us/library/ff607742(v=office.15)`.

Site Collections and Sites

The site is the core artifact to SharePoint and represents the starting point for developers; that is, you can't start developing until you have created a site collection. A variety of site templates are available for you to use. Figure 1-8 shows a selection of default templates from which you can choose when creating a new site collection. This example includes some of the choices available for creating a new site collection within an Office 365 instance, but a similar set of templates are available within SharePoint Foundation and Server. The ones in Figure 1-8 are only a subset of those available. To view the other ones, when creating a new site collection in the new site collection dialog click the Meetings, Enterprise, Publishing, or Custom tabs to see more. Each of these tabs contains specific templates that you can use for those purposes — for example, managing meetings, blogs, short-term document workspaces, longer-term projects, and, of course, building custom templates.

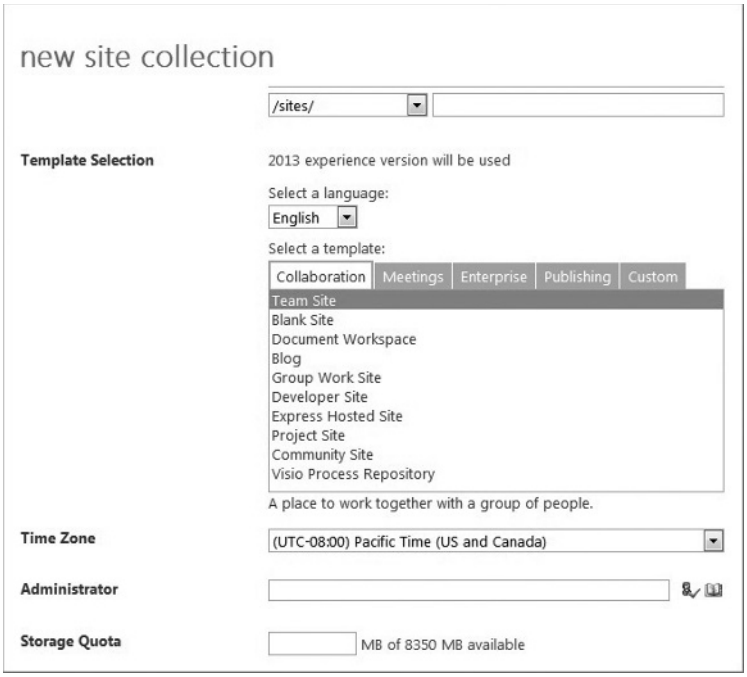

FIGURE 1-8

Because you need a SharePoint site as a starting point, let's first go ahead and create a SharePoint site. This exercise assumes you have an Office 365 tenancy up and running. At the time of writing, you could go to: `http://www.microsoft.com/office/preview/en` and click the Try button, and then under the Enterprise category click Try. You'll then be guided through a short wizard to provision an Office 365 instance.

TRY IT OUT Creating Your First SharePoint Site

To create a simple Team site within your Office 365 instance:

1. Navigate to the administration portal of your Office 365 portal: `https://portal.microsoftonline.com/admin/default.aspx`. Enter your Office 365 user ID (for example, `superme@mydomain.onmicrosoft.com`) and a password.
2. Click the Admin drop-down list and select SharePoint, which opens the SharePoint Administrator Center, (see Figure 1-9).

FIGURE 1-9

3. In the SharePoint Administration Center, click Site Collections — located on the left side of the screen.
4. Under the Site Collections tab, select New and then click Private Site Collection as shown in Figure 1-10.

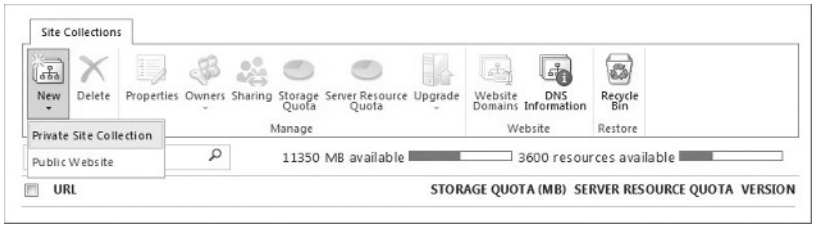

FIGURE 1-10

5. In the new site collection dialog (shown in Figure 1-11), provide a Title and a Public Website Address, select a Template (for this example choose the Developer Site under the Collaboration tab), leave the Time Zone to the default setting, add yourself as the Administrator, and provide a Storage Quota and Server Resource Quota.
6. Click OK.

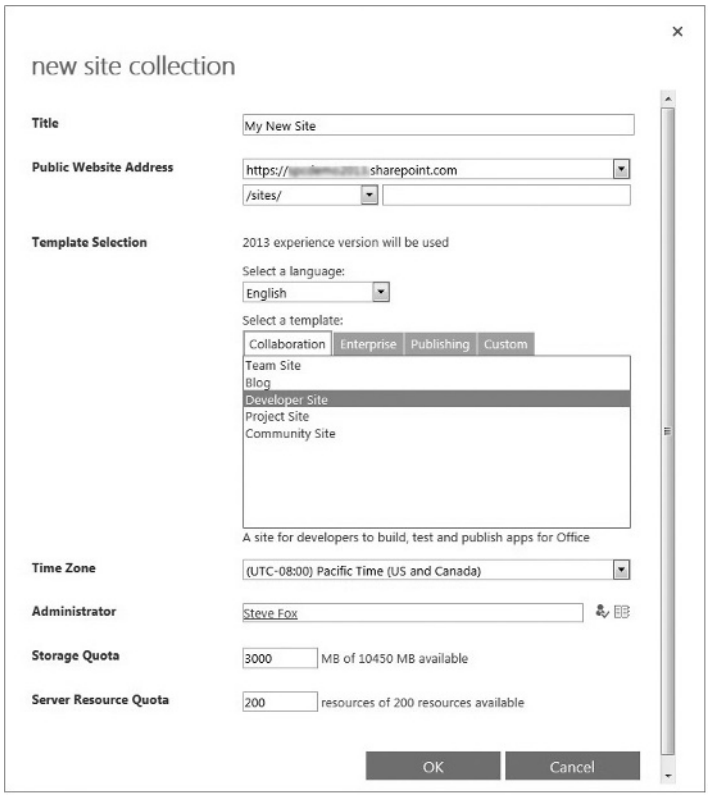

FIGURE 1-11

7. Wait a couple of minutes while Office 365 provisions the new site using the Developer Site template. When it's done, click the link to your new site, shown in Figure 1-12: `https://mydomain.sharepoint.com/sites/dev`.

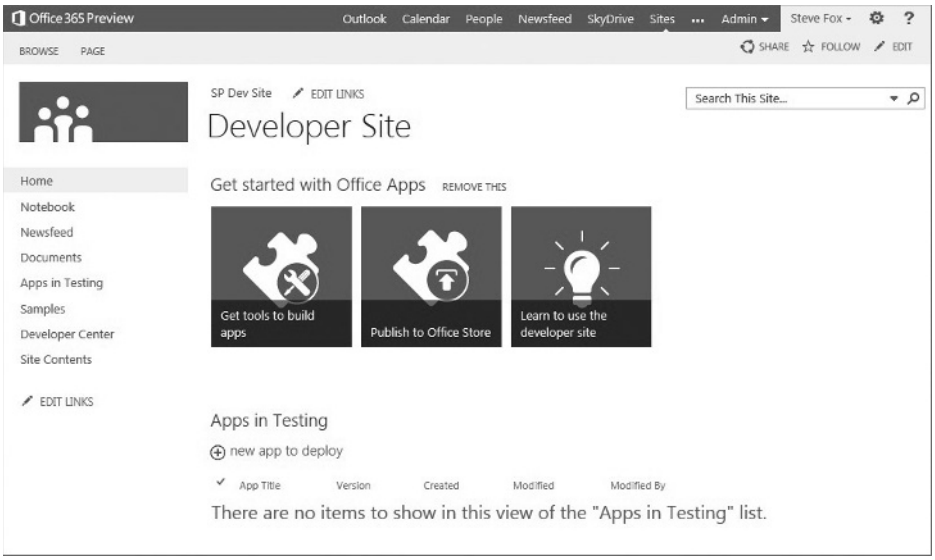

FIGURE 1-12

The new site should look similar to Figure 1-12. Go ahead and explore the site. You can click the live tiles at the top of the site, click the links on the left-hand side of the site, add subsites to this site collection, and so on.

How It Works

The baseline artifact that you created here was a site collection. The site collection in this case was a developer-specific site and represents the uppermost root site that you'll work from within SharePoint. You can now add default apps (such as lists or document libraries), create and deploy Apps for SharePoint, configure the look and feel of the site, and so on.

The site collection is a site that you can customize and interact with. You grow your SharePoint site collection by adding additional Websites to it. Any site you create underneath the site collection is called a *subsite*. This might seem confusing, but just think of the site collection being the *parent* and the sites within that collection being the *children*. This is important because by default children sites inherit the parent site's properties (such as permissions).

Creating the site collection is the most fundamental development task within SharePoint; once you've completed this, you're ready to begin building apps. To do so, it helps to understand the types of APIs that are available to you.

SharePoint 2013 APIs

After you create a new site collection, you now have the fundamental parent object in place to begin coding against. As a developer, you'll want to understand what you can do with this site now that it's created. This requires a baseline understanding of the available APIs and services. You'll want to be most familiar with two sets of object model levels: the server object model and the client-side object model.

Server Object Model

The server object model is reserved for full-instance SharePoint Foundation or SharePoint Server installations. You essentially have *carte blanche* access to the server when you install and host it yourself. It is also the broadest of the available APIs within the managed SharePoint classes. You can build many different types of applications using the server object model for tasks such as document library or list creation or manipulation, retrieving user information, site administration, backup, taxonomy and metadata management, and so on. The bulk of the server object model classes are available in the `Microsoft.SharePoint` namespace.

The server object model is available through a set of assemblies that are deployed to the global assembly cache (GAC), so you must deploy apps on the server for them to use these classes and libraries. However, you can do quite a lot with them. For example, the following code snippet sets the title and description for a list called Tasks and then calls the `Update` method to update the changes:

```
SPList myTaskList = mySPTaskSite.Lists["Tasks"];
myTaskList.Title="Sales Task List";
myTaskList.Description="A list of sales tasks.";
myTaskList.Update();
```

Client-side Object Model

The client-side object model is also available for your use in remote or client-side applications. These applications could be .NET, Silverlight, or one of the new additions to SharePoint 2013, the mobile API. This is significant because it provides you with the ability to create and deploy apps that are not necessarily dependent on server-side resources. For example, the following code snippet shows a sampling of SharePoint client-side code. You can see right away that the client-side object model looks somewhat different; in this snippet, you're setting the context for your SharePoint site, loading it, and then calling the `ExecuteQuery()` method — which executes everything that has been set before that line of code (think of a more optimized, batch processing approach). The final line of code sets the `Text` property of the `lblSPLabel` object (a label) to be the title of the SharePoint site.

```
ClientContext context = new ClientContext("http://MySharePointSite");
Web web = context.Web;
context.Load(web);
context.ExecuteQuery();
lblSPLabel.Text = web.Title;
```

JavaScript Object Model

SharePoint 2013 also has a JavaScript object model. This is an extension to what is available in the client-side object model and provides an opportunity for you to build a broad variety of SharePoint-hosted apps that can further integrate with HTML5, JQuery, and other Web technologies.

Moving Beyond the Models

Beyond the server object model and client-side object model, many other ways exist that you can build applications and solutions for SharePoint. For example, you can use a rich set of OData and REST (Representational State Transfer) services to interact with SharePoint data. Note also that the client-side object model has many REST counterparts to ensure you have multiple ways to build your Web apps. The REST services within SharePoint support both Atom and JSON formats.

Within each SharePoint site that you create, you're going to find many different opportunities to create and program against data. In the world of SharePoint, *data* can mean many different things, such as:

- Integrating with Access Services
- Interacting with SQL Server data
- Interacting with service endpoints through BDC to integrate with LOB and non-Microsoft systems
- Leveraging SQL Server Reporting Services or PerformancePoint Server to bring enhanced BI into your solutions
- Coding against data that might come from a SharePoint list where users manually enter the list data, and you programmatically code against it

To help with data programmability, you can use both the server- and client-side object models, but WCF Data Services are also supported within SharePoint. This enables you to interact with data through a LINQ provider and use LINQ syntax in .NET or Silverlight applications. For example, you can target both `listdata.svc` for list data or `client.svc` for accessing SharePoint entities beyond list data.

The preceding APIs represent a core set of ways in which you can program against SharePoint — from the fully self-hosted server instance to the cloud-hosted Office 365. Beyond these core APIs and services, you'll find you can programmatically interact with many of the services that ship with SharePoint Foundation or Server. You'll also find that you can build and deploy cloud-hosted apps (whether to Windows Azure or to other domains or Web technologies).

Many of you who will develop for SharePoint may also administer certain aspects of your SharePoint site. This might mean that you have to install and configure SharePoint, understand how to upgrade some of your solutions from SharePoint 2010 to 2013, or even create new Web applications or sites using the Central Administration site functions. Because cases may occur where you

want to leverage the capabilities built into SharePoint Central Administration, the following section provides an overview of interacting with SharePoint 2013 in this manner.

SHAREPOINT CENTRAL ADMINISTRATION

Although this book is not on administration, it is worth having a high-level introduction to the topic. After you install SharePoint 2013 (Foundation or Server), a separate site collection is created for your use for performing the different administrative functions that you might do on a daily basis. This site collection is called the Central Administration site. This site collection is run as its own Web application in IIS and is separate from the site collections you create, but it is still the central point of administration for your SharePoint site. All farm server administrators can access this site, and, much like your regular SharePoint sites, you can edit and customize the Central Administration site. Figure 1-13 shows the SharePoint Central Administration site.

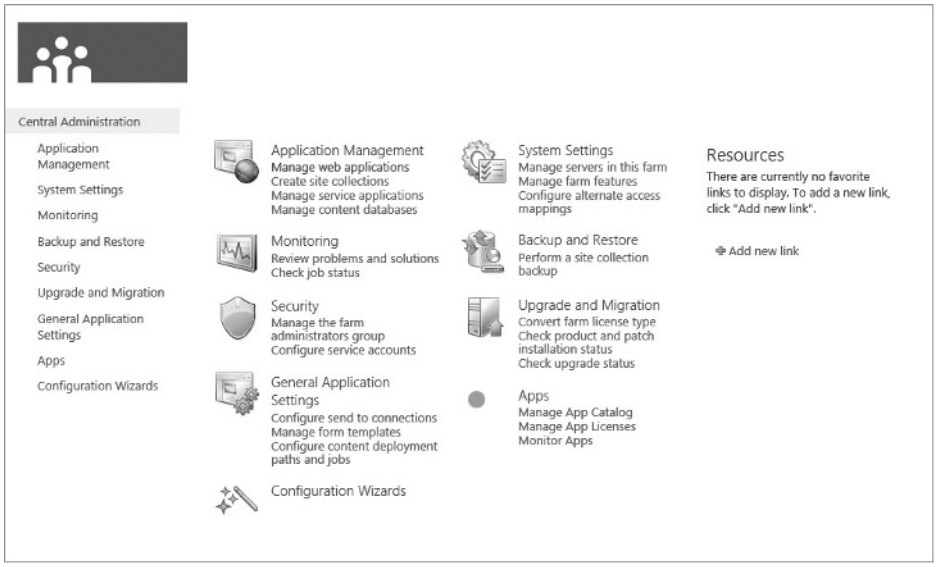

FIGURE 1-13

If you sign up for an Office 365 instance, you also have an administration site that you will certainly use. You saw this already in the exercise you walked through earlier, and in Figure 1-14 you can see a variety of site collection administration features — including BCS content type management, profile management, term store management, and search management, among others.

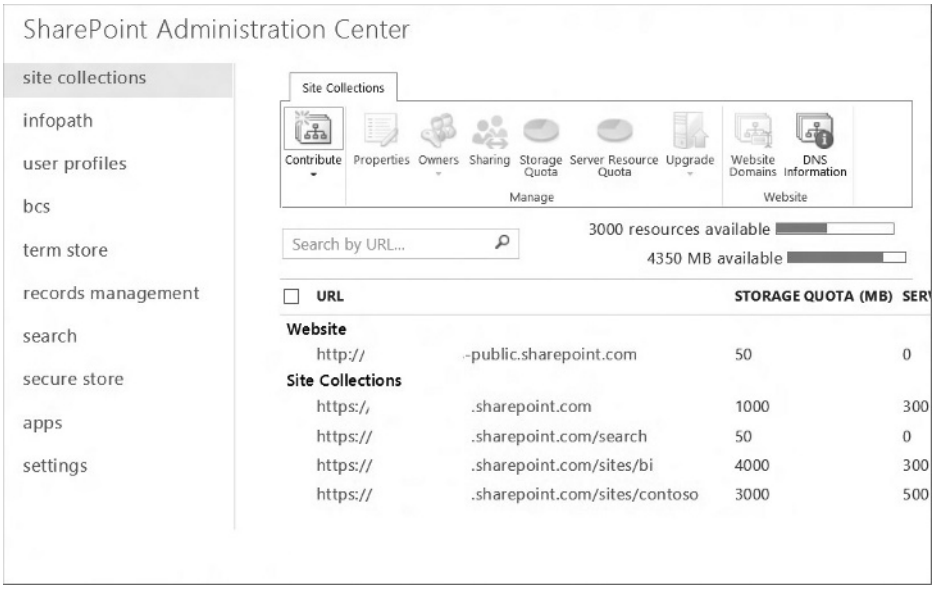

FIGURE 1-14

Within these administrative features you can manage a number of activities, which are broken out into the following nine areas:

- Application management
- Monitoring
- Security
- General application settings
- System settings
- Backup and restore
- Upgrade and migration
- Configuration wizard
- Apps

The following sections explain how to use the Central Administration site to manage activities across all of these nine areas.

Application Management

Application Management is the place where you can accomplish tasks such as create new Web applications and site collections, and, more generally, manage the services that are installed on your SharePoint site (for example, Excel Services or BCS) and manage your content database. Using the application management options, you can accomplish tasks such as modify the properties of the content database, activate features, create new site collections, and so on.

> ***NOTE*** *The content database is a SQL Server that stores SharePoint data, and is the reason why SharePoint takes a dependency on SQL Server upon installation.*

Monitoring

Monitoring is the central place within Central Administration to manage reporting, monitoring, and the status of your SharePoint site. The Monitoring site contains three areas:

- **Health status**: Health status provides a place for you to see the status of different services on your SharePoint Server (such as Visio services or farm-level services). You can see which services are failing, for example, through reports you access in this area. Health status also enables you to define rules (such as the scheduling of application pool recycles).
- **Timer jobs:** Timer jobs enable you to define specific jobs to run and when to run them (such as search crawl log cleanup or audit log trimming jobs).
- **Reporting**: Reporting provides you with a set of tools that enables you to create and manage reports, run diagnostic logging, and view reports on various server-side activities.

Security

Security covers a number of areas, including the management of administrator accounts, the configuration and management of service accounts, the management of password change settings and policies, and the specifications of authentication providers, trusted identity providers, antivirus settings, blocked file types, self-service security, and secure token services. The security settings in this area supplement the security in the main browser UI, where users and site administrators can assess specific permissions that relate to users for their sites.

General Application Settings

The General Application Settings site is where you configure a number of general options for your SharePoint site collections and sites. For example, you'll often find that you want to have the capability for your SharePoint site to send mail to users. You configure these options from within this part of the site.

Also, in the context of WCM, you might want to manage a number of deployment and approval options (such as content deployment location and approvers of that content). You also manage that type of activity from within the General Application Settings.

In general, think of this site as the generic settings for your SharePoint sites.

System Settings

Converse to using the SharePoint site settings, you might also want to configure more server-centric settings such as farm-level or access features, or even manage the services (for example, Excel Services) that are available to users of the site collection. You manage these types of settings from within the System Settings site.

Backup and Restore

At some point, you might find that you must back up and restore your SharePoint site. The backup and restore features within Central Administration enable you to create and schedule regular backups for your SharePoint, perform ad hoc backups, restore from a previously backed-up SharePoint site, and so on. Essentially, this is your point of entry if you want to ensure that you have a failover plan for backing up a site.

Although you might think you'll never need to use the backup and restore features, sometimes heightened permissions sets converge with mistakes, which often result in new users deleting parts of a site by accident — which might include something you've created as a developer.

Upgrade and Migration

At some point, you might find yourself wanting to upgrade from one version of SharePoint to another — for example, moving from SharePoint Standard to SharePoint Enterprise. This requires a license and some facility to upgrade the server.

You can do this type of action from within the Upgrade and Migration part of the Central Administration site. Note that you can also install service patches and check on installation and upgrade progress from within this part of the administration toolset.

Configuration Wizard

The Configuration Wizard is simply a step-by-step wizard that configures your SharePoint server for you. You should have seen this wizard when you first installed SharePoint. However, if you want to run it again after installation to change some of the configurations on your SharePoint server, you can do so.

Apps

Apps is a new category within the Central Administration site that enables you to manage different facets of the apps that are installed on your SharePoint instance. For example, you can use Apps to manage the licenses, ensure that apps are running and performing in an error-free way, and also manage the App Catalog.

SUMMARY

This chapter provides a first look at SharePoint — both for those who have never seen it and for those who are experienced SharePoint developers.

In this chapter, SharePoint is broadly defined as a business productivity platform for the enterprise and the Internet. More specifically, for the developer (and in the context of this book), you should see SharePoint as a platform that supports developer productivity, has extensive platform services, and can support multiple deployment options. With SharePoint you can leverage an abundance of APIs, a rich server and client-side object model, and a powerful set of services to create some very compelling applications. A great set of tools is also available that will support your efforts at evolving or improving your SharePoint development skills.

EXERCISES

Answers to Exercises can be found in Appendix A.

1. Define what SharePoint is for both the end user and the developer.
2. What are the different types of applications you can build for SharePoint 2013?
3. What are some of the key services in SharePoint 2013?
4. What are the two different types of object models in SharePoint, and how might you use them?
5. Create a new SharePoint site using the Team site template. Add a new list and document library. Add list items to the list and add documents to the document library.

▶ WHAT YOU LEARNED IN THIS CHAPTER

ITEM	DESCRIPTION
SharePoint	Collaborative platform for many different types of organizations.
SharePoint for the Developer	SharePoint is about developer productivity, the availability of rich platform services, and the capability to manage and deploy your applications with maximum flexibility.
SharePoint Foundation	Core edition for SharePoint. It ships as a free download.
SharePoint Server	The Enterprise edition of SharePoint (full-featured) referred to as SharePoint throughout the book.
Office 365	Cloud-hosted version of Office and SharePoint that provides you with a rich version of SharePoint both for collaboration and development.
SharePoint Central Administration	The site collection that you use to administer your SharePoint site.
SharePoint Administration Center	The administration site for Office 365.

RECOMMENDED READING

SharePoint 2013 Developer Overview — `http://msdn.microsoft.com/en-us/library/jj164084(v=office.15).aspx`

TechNet article on API updates — `http://technet.microsoft.com/en-us/library/ff607742(v=office.15)`