200 XP

Understand Office add-ins fundamentals

14 minutes

The Office add-ins platform enables you to extend the functionality of Office applications. In this unit, you'll explore various ways you can use add-ins to extend and interact with Office applications. You'll also learn about configuring your add-in using the add-in's manifest file.

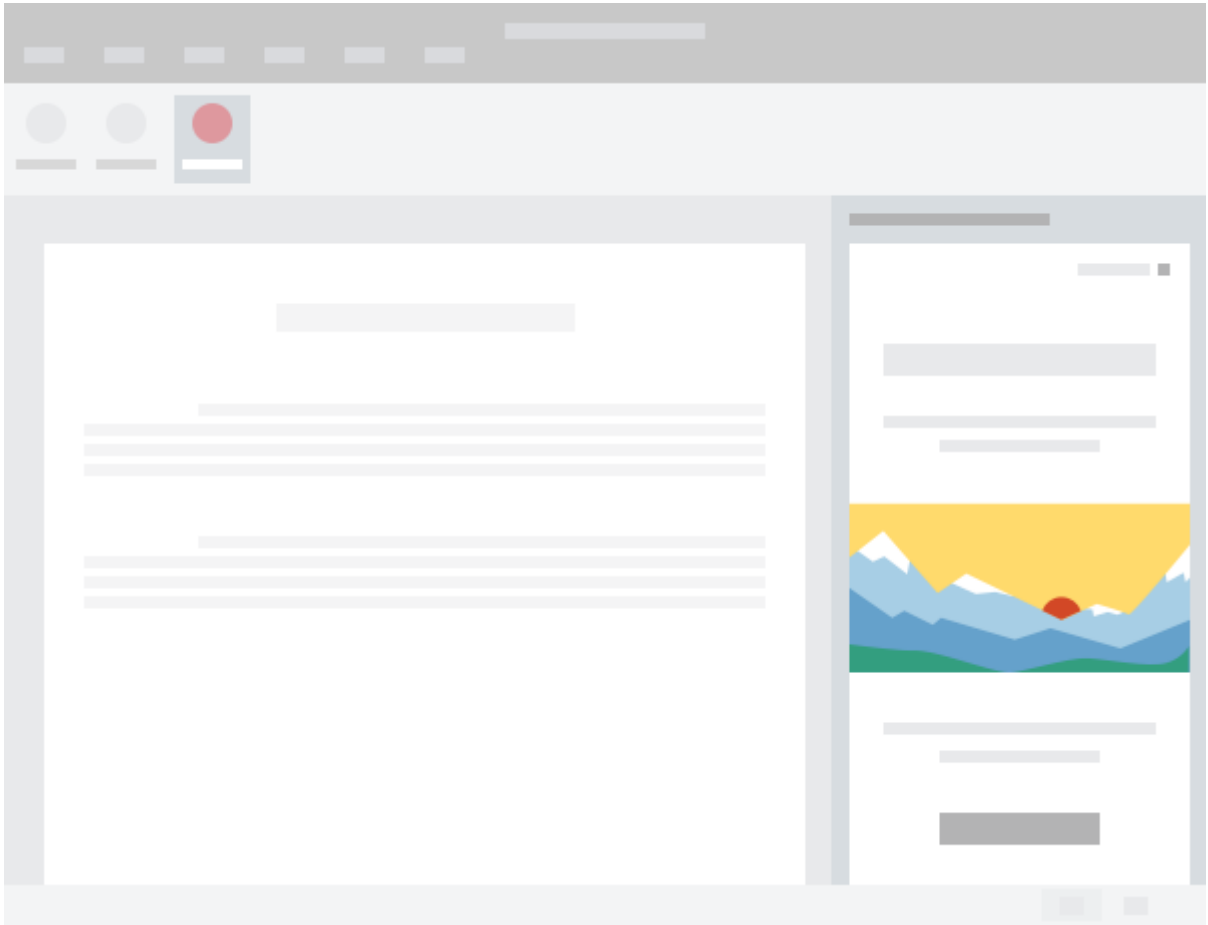
Understand task pane and content add-ins

Office add-ins provide several options for how your solution can interact with an Office application. In this unit, we discuss two of those options:

- Task pane
- Content

Task pane add-ins

Task pane add-ins allow user interaction through a panel displayed within an Office application. Through the task pane interface, you can enable the user to modify documents or emails, view data from a data source, and more. In the following image, the task pane is the panel that's displayed to the right of the document.



Task pane add-in displayed within an Office application

In newer versions of Word, Excel, and PowerPoint, you can configure the task pane to be displayed automatically when a user opens a file. The user will need to have your add-in installed first to activate this behavior.

Define the task pane add-in type

As described previously, an add-in's manifest file defines the settings and capabilities of the add-in.

To configure an add-in as a task pane add-in for any Office application except Outlook, set the `xsi:type` attribute to `TaskPaneApp` within the `OfficeApp` element of the manifest file, as shown in the following example.

XML Copy

```xml
<OfficeApp
  ...
  xsi:type="TaskPaneApp">
   ...
</OfficeApp>
```

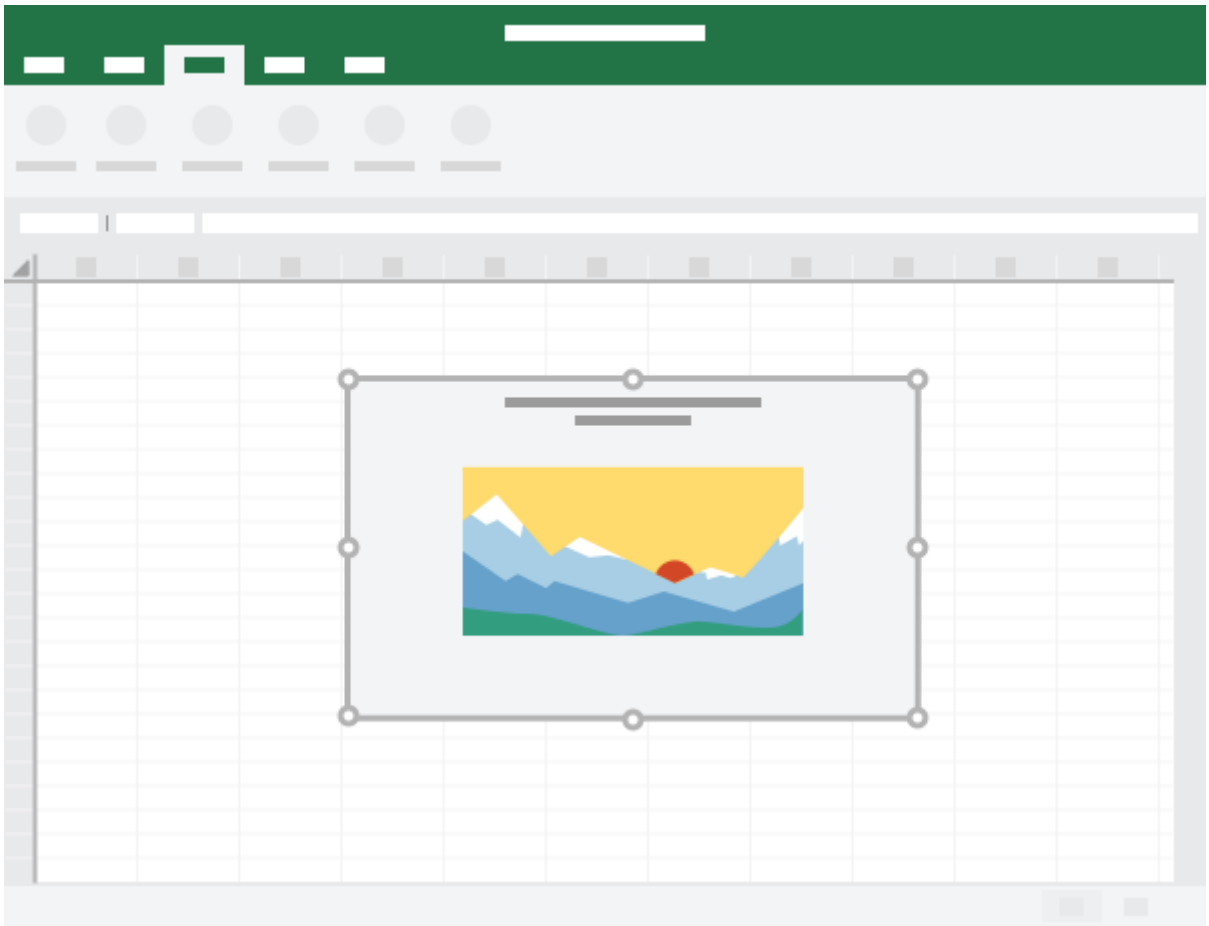
To configure an add-in for Outlook, set the `xsi:type` attribute to `MailApp` within the `OfficeApp` element of the manifest file, as shown in the following example.

XML Copy

```xml
<OfficeApp
  ...
  xsi:type="MailApp">
   ...
</OfficeApp>
```

Content add-ins

Content add-ins can be used to insert an object into an Excel spreadsheet or PowerPoint presentation. That object can be a web-based data visualization, media, or other external content. In the following image, the content add-in is displayed near the center of the document.



Content add-in loaded within an Office application

Define the content add-in type

As described previously, an add-in's manifest file defines the settings and capabilities of the add-in. To configure an add-in as a content add-in, set the `xsi:type` attribute to `ContentApp` within the `OfficeApp` element of the manifest file, as shown in the following example.

XML Copy

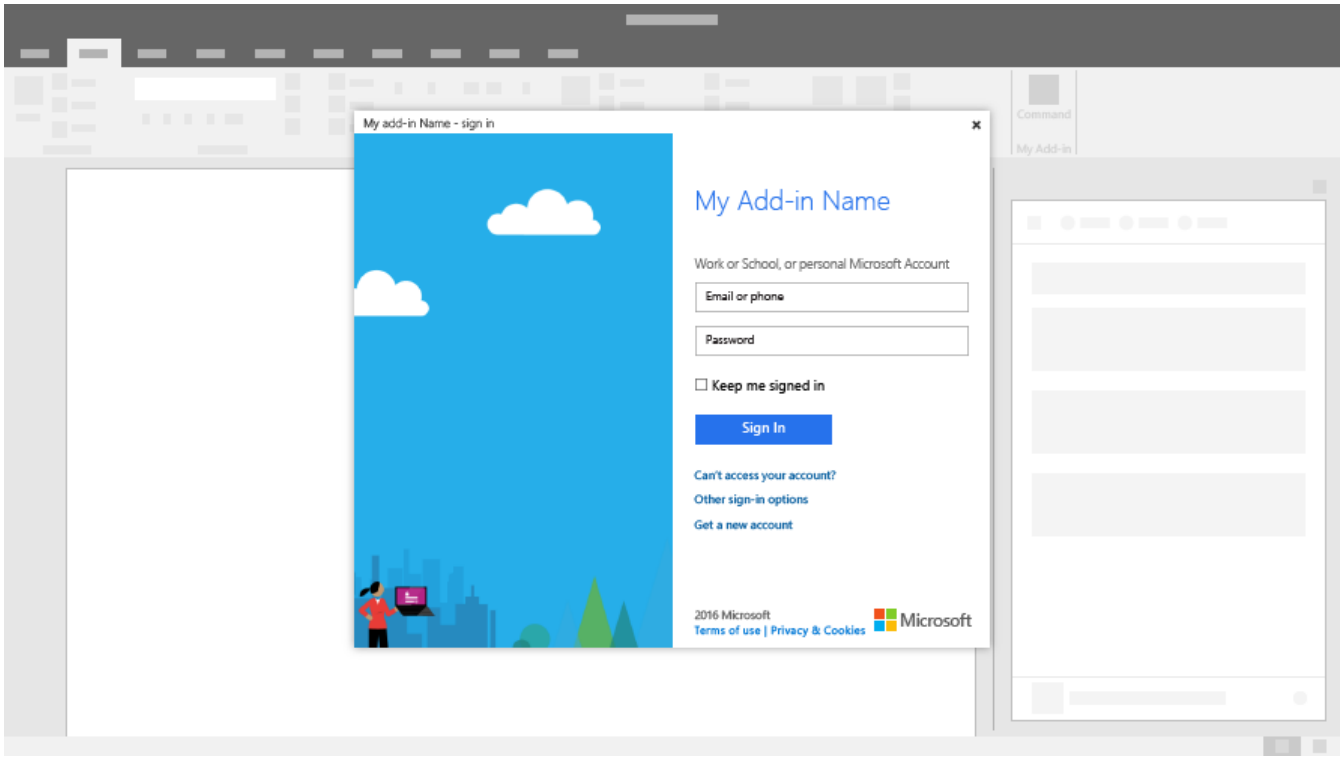
```xml
<OfficeApp
  ...
  xsi:type="ContentApp">
   ...
</OfficeApp>
```

Understand Office add-ins dialogs

The Office add-ins platform enables you to display a dialog for your users to:

- Sign into an integrated service (for example, authenticate with Microsoft Account, Google, or Facebook).
- Confirm the user's action.
- Run a task that might be too confined in a task pane (for example, view a video).

The dialog window isn't modal, meaning that your user can continue to interact with the Office application and your add-in while the dialog window is displayed. The following image shows a dialog being displayed in an Office application.

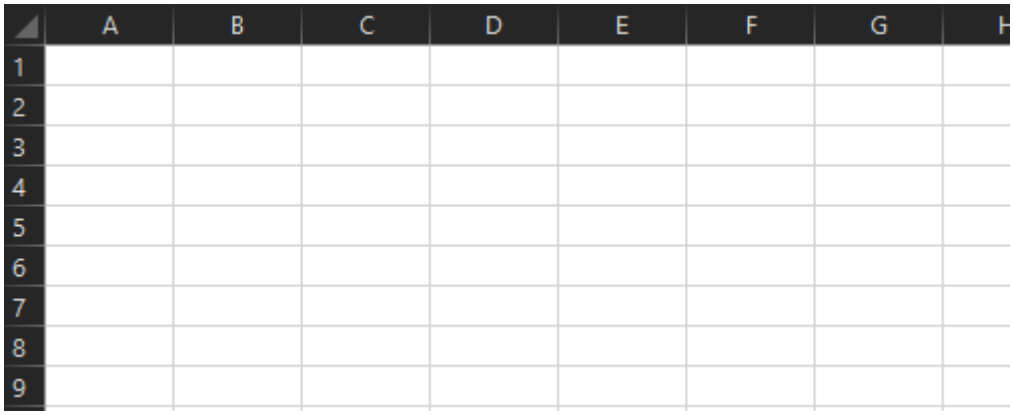


Dialog displayed in an Office application

Understand Office add-ins custom functions

Office add-ins enable you to create custom JavaScript or TypeScript functions that can be accessed like built-in Excel functions such as `SUM()`.

The following image shows a custom function called `SPHEREVOLUME` being entered in Excel.



Custom function being entered in Excel

The following code sample shows the JavaScript code for the `SPHEREVOLUME()` function shown previously.

JavaScript Copy

```javascript
/**
 * Returns the volume of a sphere.
 * @customfunction
 * @param {number} radius
 */
function sphereVolume(radius) {
  return (Math.pow(radius, 3) * 4 * Math.PI) / 3;
}
```

Where can you use custom functions?

Custom functions are available in Excel on the following platforms.

- Windows *(connected to an Office 365 subscription)*
- macOS *(connected to an Office 365 subscription)*
- Web browser

Define the custom function add-in type

To configure an add-in to contain custom functions, the key settings in the manifest are as follows for Excel add-ins.

XML Copy

```xml
<OfficeApp
  ...
  xsi:type="TaskPaneApp">
   ...
  <Hosts>
    <Host Name="Workbook"/>
  </Hosts>
   ...
  <VersionOverrides
xmlns="http://schemas.microsoft.com/office/taskpaneappversionoverrides"
xsi:type="VersionOverridesV1_0">
    <Hosts>
      <Host xsi:type="Workbook">
        <AllFormFactors>
          <ExtensionPoint xsi:type="CustomFunctions">
            ...
          </ExtensionPoint>
        </AllFormFactors>
      </Host>
    </Hosts>
   ...
  </VersionOverrides>
</OfficeApp>
```

Understand add-in commands

Add-in commands are UI elements that extend the Office UI and start actions in your add-in. You can use add-in commands to add a button on the ribbon or an item to a context menu. When users select an add-in command, they start actions such as running JavaScript code, or showing a page of the add-in in a task pane. Add-in commands help users find and use your add-in, which can help increase your add-in's adoption and reuse, and improve customer retention.

Add-in commands in Excel, Word, PowerPoint, and OneNote

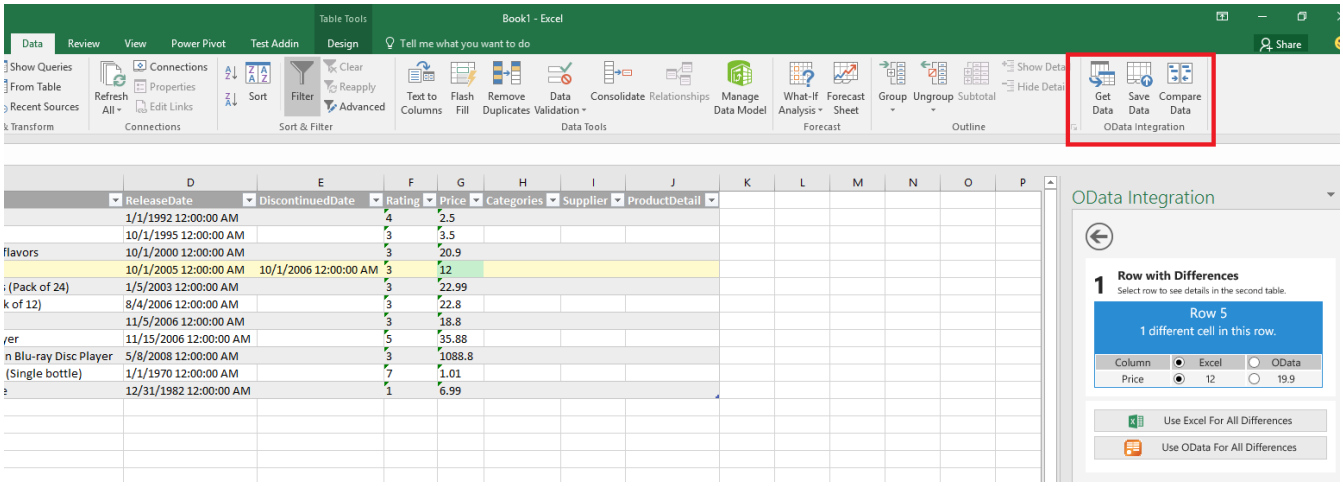
You can configure an add-in so that a user can run it by selecting:

- Office application's ribbon or command overflow menu button
  - Key manifest setting: `<ExtensionPoint xsi:type="PrimaryCommandSurface">`.
- Context menu item
  - Key manifest setting: `<ExtensionPoint xsi:type="ContextMenu">`.

An add-in command can also open a submenu with more commands.

> **Note**
>
> Content add-ins don't currently support add-in commands.

The following image shows three add-in commands (custom buttons) added to the **Data** tab of the Excel ribbon.



Add-in commands in Excel on Windows

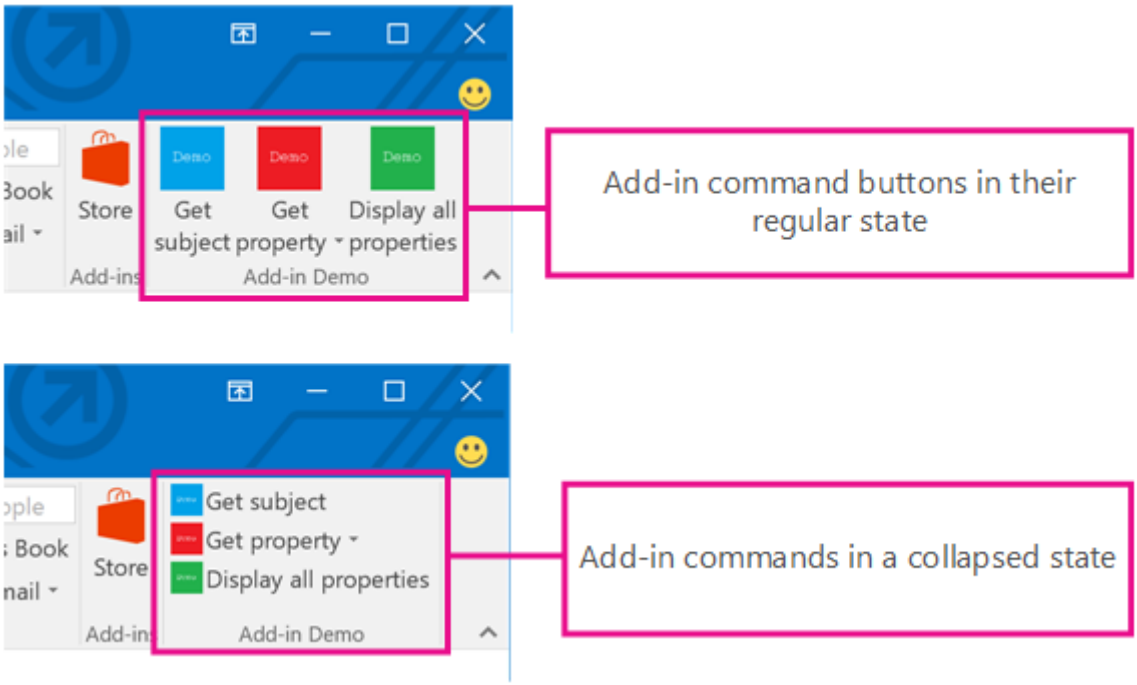
Add-in commands in Outlook

You can configure an add-in so that a user can run it by selecting a button in the Office ribbon or command overflow menu when the user is:

- Reading a message in the reading pane or in a pop-out window.
  - Key manifest setting: `<ExtensionPoint xsi:type="MessageReadCommandSurface">`.
- Composing a message.
  - Key manifest setting: `<ExtensionPoint xsi:type="MessageComposeCommandSurface">`.
- Creating or viewing an appointment or meeting as the organizer.
  - Key manifest setting: `<ExtensionPoint xsi:type="AppointmentOrganizerCommandSurface">`.
- Viewing a meeting as an attendee.
  - Key manifest setting: `<ExtensionPoint xsi:type="AppointmentAttendeeCommandSurface">`.

An add-in command can also open a submenu with more commands.

The following images show three add-in commands (custom buttons) added to the ribbon in Outlook. In the first image, the buttons are rendered in a regular state; in the second image, the buttons are rendered in a collapsed state.



Add-in commands in Outlook on Windows

Where can you use add-in commands?

Add-in commands are available in Excel, Outlook, OneNote, PowerPoint, and Word as shown in the following table.

Platform	Major Office version	Subscription or one-time purchase?	Notes
Windows	*Not applicable*	connected to Office 365 subscription	*Not available in OneNote*
	2019	one-time purchase	*Not available in OneNote*
	2016	one-time purchase	*Only available in Outlook on Exchange 2016 (requires post-release update) or later. Not available in Office other applications.*
	2013	one-time purchase	*Only available in Outlook on Exchange 2016 or later. Requires post-release updates for Outlook and Exchange 2016. Not available in other Office applications.*
macOS	*Not applicable*	connected to Office 365 subscription	*Not available in OneNote*

Platform	Major Office version	Subscription or one-time purchase?	Notes
	2019	one-time purchase	*Not available in OneNote*
	2016	one-time purchase	*Not available in OneNote*
iOS	*Not applicable*	connected to Office 365 subscription	*Only available in Outlook*
Android	*Not applicable*	connected to Office 365 subscription	*Only available in Outlook*
web browser	*Not applicable*	*Not applicable*	*Available in all supported Office applications*

Understand the purpose of the add-in manifest

An Office add-in's XML manifest file defines the settings and capabilities of the add-in. You can configure it to control how your add-in is rendered and behaves in the targeted Office applications.

What the manifest defines

In the manifest, you define key information about the add-in, including:

- Add-in metadata (for example, ID, version, description, display name, default locale)
- Information about how the add-in integrates with Office (for example, target applications, custom functionality, add-in commands)
- Location of images the add-in should use for branding and command iconography
- Permissions that the add-in requires
- Dimensions of the add-in (for example, default dimensions for content add-ins, requested height for Outlook add-ins)
- Rules that specify when the add-in should activate in a message or appointment (Outlook only)

How the manifest is used

An add-in manifest is used in the following ways:

- The Office applications where your add-in runs use information from the manifest to render add-in UI and wire up custom buttons or menu entries.
- If you publish your add-in to AppSource:
  - Information from the manifest (name, description, author, logo, and so on) is used to create the app entry that's displayed to potential customers in AppSource.
  - The AppSource validation process reads information from the manifest and validates that your add-in runs on expected platforms.

Summary

The Office add-ins platform enables you to extend the functionality of Office applications. In this unit, you explored various ways you can use add-ins to extend and interact with Office applications. You also learned about configuring your add-in using the add-in's manifest file.

Fundamental concepts about Office add-ins

1. Which of the following statements is true about Office add-ins?

- ○ Add-ins can run only in Office on Windows, Mac, and iPad/iOS.
- ○ Office 365 administrators can use centralized deployment to deploy add-ins across their organization.
- ○ A developer publishes their add-in to AppSource to hide it from the general public.

2. The user needs to sign in to access add-in functionality within an Office application. Which Office add-in platform feature should the developer display to facilitate the sign-in process?

- ○ add-in command
- ○ custom function
- ○ dialog

3. What file defines the settings and capabilities of an Office add-in?

- ○ dialog file
- ○ manifest file
- ○ web.config file

Check your answers