

TOPICS Accessibility ▾

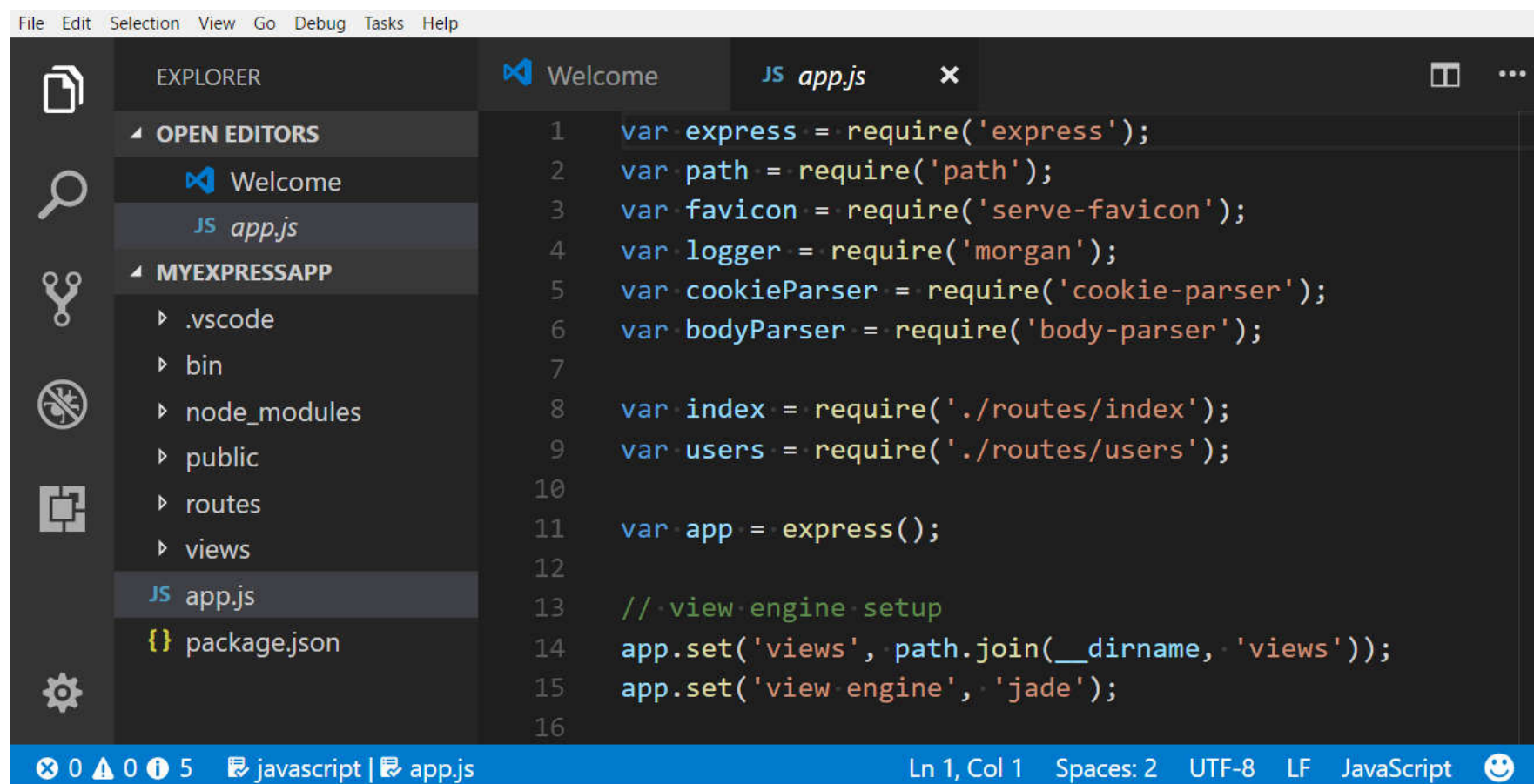
Accessibility

<https://github.com/Microsoft/vscode-docs/blob/master/docs/editor/accessibility.md>

Visual Studio Code has many features to help make the editor accessible to all users. Zoom and High Contrast colors improve editor visibility, keyboard-only navigation allows use without a mouse, and the editor has been optimized for screen readers.

Zoom

You can increase the Zoom level in VS Code with the **View > Zoom In** command (`Ctrl+=`). The zoom level increases by 20% each time the command is executed. The **View > Zoom Out** (`Ctrl+-`) command lets you decrease the Zoom level.

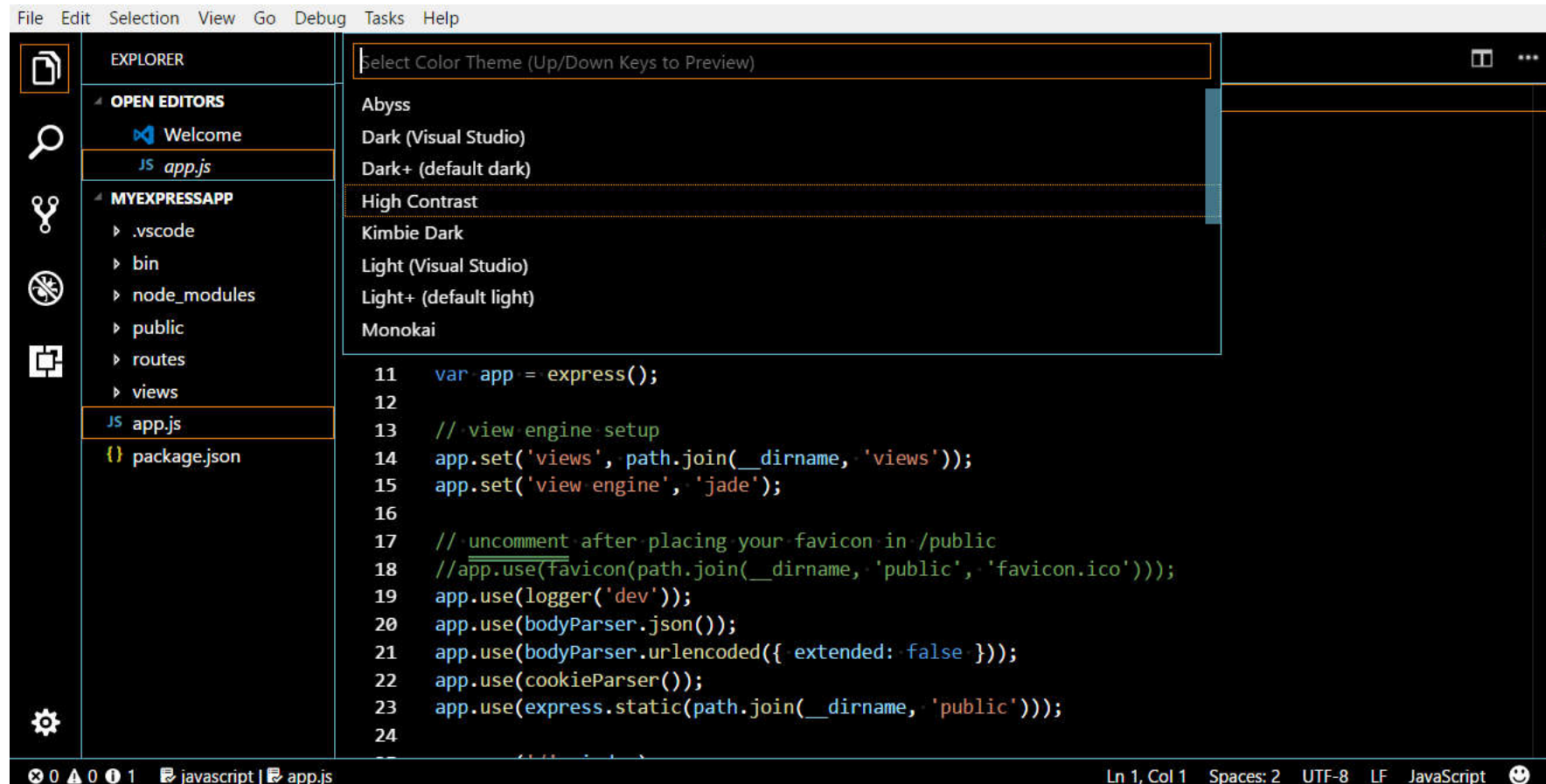


Persisted Zoom Level

When you adjust the zoom level with the **View > Zoom In / Out** commands, the zoom level is persisted in the `window.zoomLevel` setting (/docs/getstarted/settings). The default value is 0 and each increment increases the zoom level by 20%.

High Contrast theme

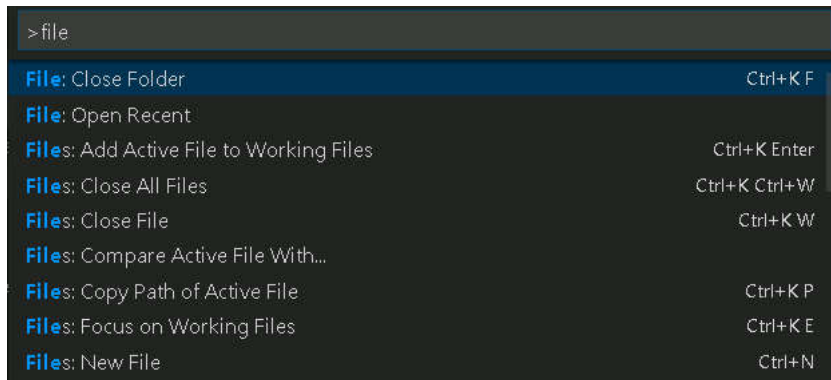
We support a High Contrast color theme on all platforms. Use **File > Preferences > Color Theme** (**Ctrl+K Ctrl+T**) to display the **Select Color Theme** drop-down and select the **High Contrast** theme.



Keyboard navigation

You will find that VS Code provides an exhaustive list of commands in the **Command Palette** (**Ctrl+Shift+P**) so that you can run VS Code without using the mouse. Press **Ctrl+Shift+P** then type a command name (for example 'git') to filter the list of commands.

VS Code also has many preset keyboard shortcuts for commands. These are displayed to the right of the command in the **Command Palette**.



You can also set your own keyboard shortcuts. **File > Preferences > Keyboard Shortcuts** (`Ctrl+K Ctrl+S`) brings up the Keyboard Shortcuts editor where you can discover and modify keybindings for VS Code actions. See Key Bindings (/docs/getstarted/keybindings) for more details on customizing or adding your own keyboard shortcuts.

Tab navigation

You can use the `Tab` key to jump between VS Code UI controls. Use `Shift+Tab` to tab in reverse order. As you tab through the UI controls, you can see an indicator around the UI element once the element gains focus.

Some areas that support Tab navigation are:

- The View switcher (File Explorer, Search, Source Control, Debug, Extensions)
- The header of collapsible sections in a view to expand/collapse
- Actions in views and sections
- Actions for items in the tree

Tab trapping

By default, pressing the `Tab` within a source code file inserts the Tab character (or spaces depending on your Indentation setting) and does not leave the open file. You can toggle the trapping of `Tab` with `Ctrl+M` and subsequent `Tab` keys will move focus out of the file. When default `Tab` trapping is off, you will see an indicator in the Status Bar.



You can also toggle `Tab` trapping from the **Command Palette** (`Ctrl+Shift+P`) with the **Toggle Tab Key Moves Focus** action.

Read-only files never trap the `Tab` key. The **Integrated Terminal** panel respects the `Tab` trapping mode and can be toggled with `Ctrl+M`.

Screen readers

VS Code supports screen readers in the editor using a strategy based on paging the text. We have tested using the NVDA screen reader (<https://www.nvaccess.org>), but we expect all screen readers to benefit from this support.

There is a community developed NVDA add-on for VS Code (<https://github.com/pawelurbanski/nvda-for-vs-code>), that improves unintentional switching between forms and browse mode as well as providing better text reading while using IntelliSense. The add-on requires VS Code version 1.33 or higher. See the add-on README (<https://github.com/pawelurbanski/nvda-for-vs-code/blob/master/README.md>) file for more details.

When using NVDA on Windows, we recommend to update to NVDA 2017.3 or higher. NVDA 2017.3 increases NVDA's timeout for receiving a caret move event from 30ms to 100ms. This version is the first one where the built-in timeout is increased from 30ms to 100ms (<https://github.com/nvaccess/nvda/pull/7201>).

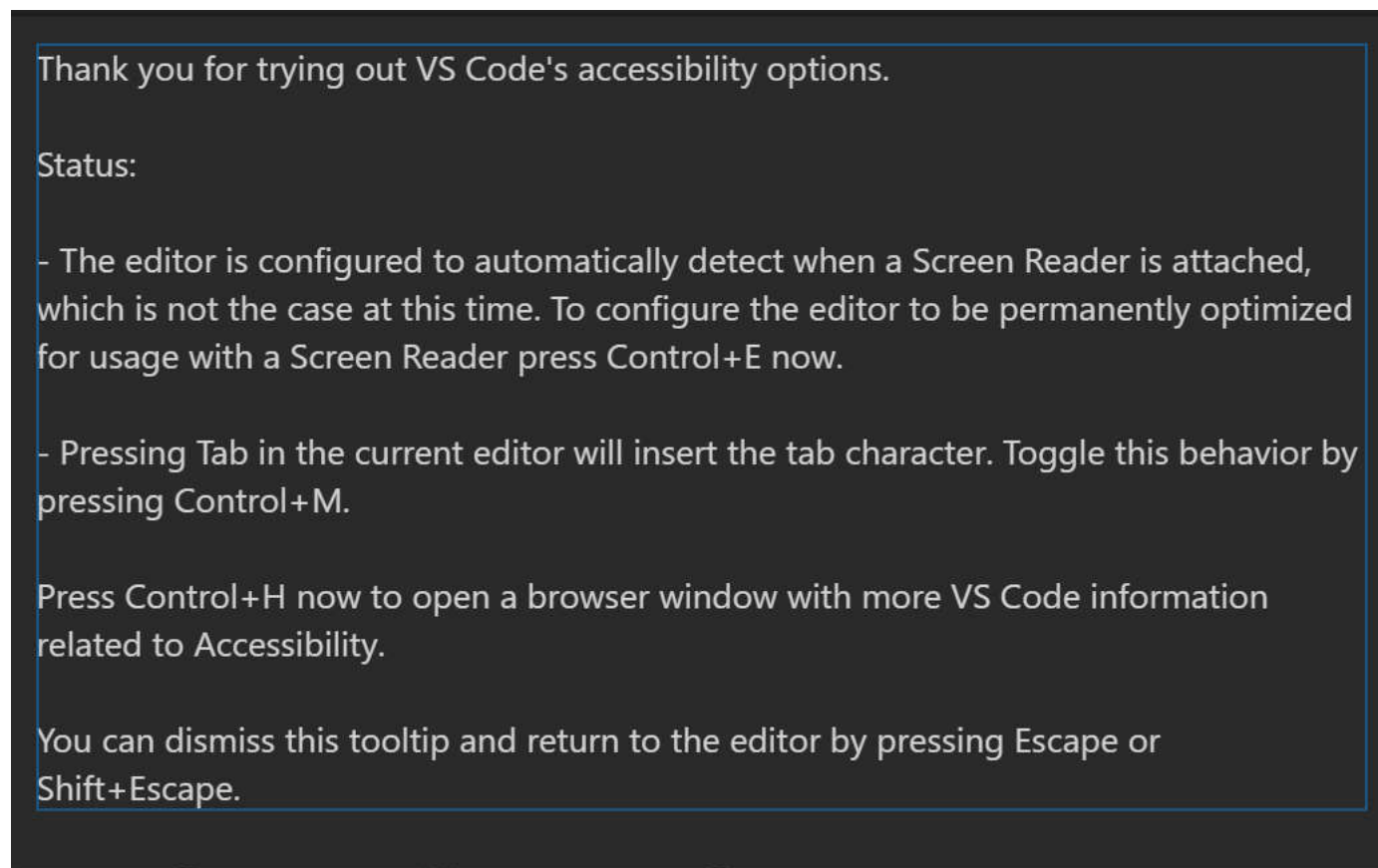
The **Go to Next/Previous Error or Warning** actions (`F8` and `Shift+F8`) allow screen readers to announce the error or warning messages.

When the suggestions pop up, they will get announced to screen readers. It is possible to navigate the suggestions using `Ctrl+Up` and `Ctrl+Down` , you can dismiss the suggestions with `Shift+Escape` and if suggestions get in your way, you can disable the auto-popup of suggestions with the `editor.quickSuggestions` setting.

The **Go to Next/Previous Difference** actions (`F7` and `Shift+F7`), when in a diff editor pane, will bring up the Diff Review pane, which allows the navigation of the diffs, presented in a unified patch format. Arrow Up and Arrow Down can be used to navigate through the unchanged, inserted or deleted lines. Pressing `Enter` will return focus to the modified pane of the diff editor at the selected line number (or closest still existing line number in case a deleted line is selected). Use `Escape` or `kb(Shift+Escape)` to dismiss the Diff Review pane.

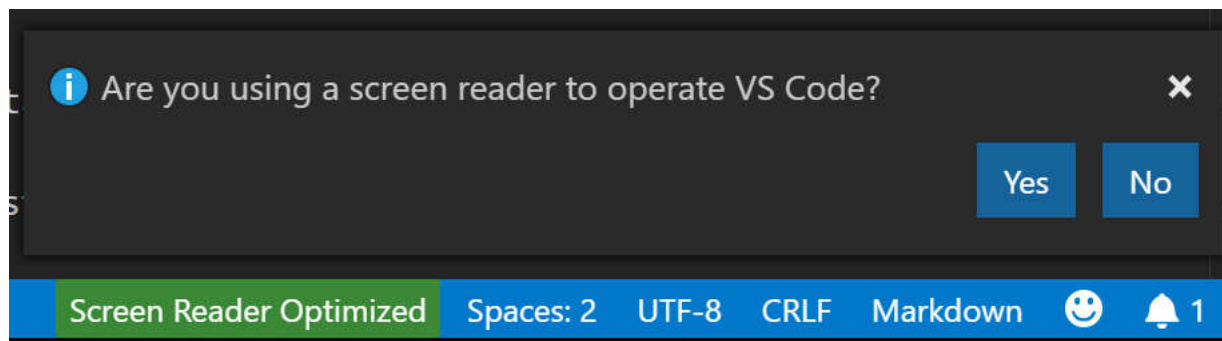
Accessibility help

You can press `Alt+F1` to trigger the **Show Accessibility Help** dialog while in an editor to check the state of various accessibility options in VS Code:



Screen reader mode

When VS Code detects that a screen reader is being used, it goes into screen reader optimized mode for the UI such as the editor and Integrated Terminal. The Status Bar displays **Screen Reader Optimized** in the lower right and you can exit screen reader mode by clicking on the display text.



Certain features such as folding, minimap (code overview), and word wrap are disabled when in screen reader mode. You can control whether VS Code uses screen reader mode with the **Editor: Accessibility Support** setting (`editor.accessibilitySupport`) and the values are `on`, `off`, or the default `auto` to automatically detect a screen reader through querying the platform.

Terminal accessibility

Output in the Integrated Terminal can be navigated through by using the scroll commands available in the Command Palette (press `F1` and search for "Terminal Scroll").

Debugger accessibility

The VS Code debugger UI is user accessible and has the following features:

- Changes in debug state are read out (for example 'started', 'breakpoint hit', 'terminated', ...).
- All debug actions are keyboard accessible.
- Both the Debug View and Debug Console support Tab navigation.
- Debug hover is keyboard accessible (`Ctrl+K Ctrl+I`).
- Keyboard shortcuts can be created to set focus to each debugger area.

Current known issues

VS Code has some known accessibility issues depending on the platform.

macOS

There is limited screen reader support for the editor with VoiceOver.

Linux

There is no screen reader support for the editor. This is because there is no accessibility implementation for Chrome on Linux.

Next steps

Read on to find out about:

- Visual Studio Code User Interface (</docs/getstarted/userinterface>) - A quick orientation to VS Code.
- Basic Editing (</docs/editor/codebasics>) - Learn about the powerful VS Code editor.
- Code Navigation (</docs/editor/editingevolved>) - Move quickly through your source code.


Was this documentation helpful?

Yes No

6/5/2019


IN THIS ARTICLE

- Zoom
- High Contrast theme
- Keyboard navigation
- Tab navigation
- Tab trapping
- Screen readers
- Accessibility help
- Screen reader mode
- Terminal accessibility
- Debugger accessibility
- Current known issues
- Next steps


 [Tweet this link](https://twitter.com/intent/tweet?original_referer=https://code.visualstudio.com/docs/editor/accessibility&ref_src=twsrc%5Etfw&text=Accessibility%20in%20Visual%20Studio%20Code&tw_p=tweetbutton&url=https://code.visualstudio.com/docs/editor/accessib) (https://twitter.com/intent/tweet?original_referer=https://code.visualstudio.com/docs/editor/accessibility&ref_src=twsrc%5Etfw&text=Accessibility%20in%20Visual%20Studio%20Code&tw_p=tweetbutton&url=https://code.visualstudio.com/docs/editor/accessib

 [Subscribe\(/feed.xml\)](#)



 [Ask questions\(https://stackoverflow.com/questions/tagged/vscode\)](https://stackoverflow.com/questions/tagged/vscode)

 [Follow @code\(https://go.microsoft.com/fwlink/?LinkID=533687\)](https://go.microsoft.com/fwlink/?LinkID=533687)

 [Request features\(https://go.microsoft.com/fwlink/?LinkID=533482\)](https://go.microsoft.com/fwlink/?LinkID=533482)


 [Report issues\(https://www.github.com/Microsoft/vscode/issues\)](https://www.github.com/Microsoft/vscode/issues)

 [Watch videos\(https://www.youtube.com/channel/UCs5Y5_7XK8HLDX0SLNwkd3w\)](https://www.youtube.com/channel/UCs5Y5_7XK8HLDX0SLNwkd3w)

Hello from Seattle. Follow @code (https://go.microsoft.com/fwlink/?LinkID=533687)  **Star**  **78,938**

Support (https://support.microsoft.com/en-us/getsupport?wf=0&tenant=ClassicCommercial&oaspworkflow=start_1.0.0.0&locale=en-us&supportregion=en-us&pesid=16064&ccsid=636196895839595242)

Privacy (https://privacy.microsoft.com/en-us/privacystatement) Terms of Use (https://www.microsoft.com/en-us/legal/intellectualproperty/copyright/default.aspx) License (/License)

 (https://www.microsoft.com)

© 2019 Microsoft