Office Add-ins prompts for GitHub Copilot

Article • 06/06/2024

The Office Add-ins prompt library is a collection of prompt examples to use with GitHub Copilot when you develop Office add-ins. GitHub Copilot is an AI pair-programmer that helps you write code. It's available as an extension for many code editors, such as Visual Studio Code and Visual Studio. You can install and launch the extensions directly through your code editor.

Copy the prompts from this article and enter them into the GitHub Copilot chat to begin experimenting. You can also open the prompt list in your code editor by downloading the file from GitHub at Office Add-ins prompts for GitHub Copilot on GitHub.

> ⓘ **Important**
>
> The prompts in this article are designed to work in English with GitHub Copilot. They've been tested to ensure that Copilot returns relevant responses, but generative AI can inherently return variable responses. Always review code produced by GitHub Copilot for accuracy before publishing or distributing your add-in to users.

We encourage you to customize your own prompts for developing Office Add-ins with GitHub Copilot. To contribute your ideas to this prompt library, review the guidance for contributing to the Office Add-ins documentation through GitHub. You can also share your prompts and feedback through the survey at Office Add-ins prompts feedback.

Prompt examples

The following sections contain prompts for multiple scenarios when developing and publishing add-ins.

> **Tip**
>
> Switch out the phrases between asterisks in the prompts to experiment with related scenarios.

Guidance for getting started

code

```
Show me the typical structure of an Office Add-in project and
explain the functionality of each file. Explain the steps and
commands to get started in *Visual Studio Code*.
```

Create an add-in project

Import a document as a Word template

code

```
Create an Office JavaScript add-in for *Word* to *import a
document as a template*. List the steps to follow in *Visual
Studio Code* to create the add-in and insert code snippets in
the correct files.
```

Create a custom function in Excel

code

```
Create an Office JavaScript Add-in for *Excel* to *create a
custom function in Excel*. List the steps to follow in *Visual
Studio Code* to create the add-in and insert code snippets in
the correct files.
```

Insert graphics into a PowerPoint presentation slide

code

```
Create an Office JavaScript Add-in for *PowerPoint* to *insert
graphics into a presentation slide*. List the steps to follow in
*Visual Studio Code* to create the add-in and insert code snippets
in the correct files.
```

Implement a feature for Excel add-ins

Add a new worksheet

code

```
Add a new worksheet *at the end* using the Excel JavaScript API.
```

Get data from a table

code

```
Retrieve the data in the range *A1:B3 on the first worksheet*
using the Excel JavaScript API.
```

Create a chart

code

```
Insert a line chart titled *"My chart"* into the current worksheet
using the data *in the range A1:B3* using the Excel JavaScript API.
```

Create a custom function to conduct calculations in Excel

code

```
Create a JavaScript custom function in Excel that conducts a
calculation.
```

Handle an event

code

```
Handle an event *when selection changes in worksheet* using
Excel JavaScript API.
```

Create shapes

code

```
Create *a yellow square* shape in the worksheet using the Excel
JavaScript API.
```

Insert a copy of an existing workbook into the current one

code

```
Insert a workbook template as base64 in the current workbook using
insertWorksheetsFromBase64 Excel JavaScript API.
```

Implement a feature for Word add-ins

Insert a paragraph

code

```
Insert a paragraph with the content *"My paragraph"* at the start
of the document using the Word JavaScript API.
```

Apply a style to a paragraph

code

```
Apply the style *"Heading1"* to the first paragraph in the document
using the Word JavaScript API.
```

Change the font

code

```
Change the font formatting of *the selected document text* using
the Word JavaScript API.
```

Insert table into a document

code

```
Insert a table named "Sample table" with sample data in the
document using the Word JavaScript API.
```

Insert a content control

code

```
Insert a content control labeled *"Sample Content Control"* into the
first paragraph and *set the content control's color to red* using
the Word JavaScript API.
```

Add a comment

code

```
Insert a comment into the document selection using the Word JavaScript API
```

Insert a document into the target document at a specific location

code

```
Import a file from local storage as a template to the current
document using the Word JavaScript insertFileFromBase64 API.
```

Implement a feature for Excel, PowerPoint, or Word add-ins

Add a dialog

code

```
Show a dialog in the application *when a user clicks a button
in the add-in* using the Office JavaScript API.
```

Get an access token

code

```
Get an access token in an Office Add-in using the Office
JavaScript API to authenticate the user with external services
or APIs.
```

Guidance for publishing add-ins

Distribute an add-in to an organization

code

```
Tell me how to deploy and distribute the local Office JavaScript
add-in code to *all employees in my organization* after
development. Provide the steps to follow.
```

Distribute an add-in to the add-in store

code

```
Tell me how to deploy and distribute the local Office JavaScript
add-in code to the *add-in store* after development. Provide the
steps to follow.
```

See also

- GitHub Copilot

Collaborate with us on GitHub

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see our contributor guide.

Office Add-ins feedback

Office Add-ins is an open source project. Select a link to provide feedback:

Open a documentation issue

Provide product feedback