

# Adobe® InCopy® 2022 Scripting ReadMe

This document contains information about scripting in Adobe InCopy 2022, including:

- A summary of the InCopy scripting documentation (see [“InCopy Scripting Documentation” on page 1](#)).
- Directions for running a script (see [“Running Scripts” on page 2](#)).
- A list of InCopy sample scripts and a brief description of each (see [“Sample Scripts” on page 2](#)).
- A list of known issues in InCopy scripting (see [“Known Issues Related to InCopy Scripting” on page 4](#)).

For more information on InCopy scripting, go to the InDesign developer documentation page, <https://www.adobe.io/apis/creativecloud/indesign.html>, or visit the InCopy Scripting User-to-User forum, [http://forums.adobe.com/community/incopy/incopy\\_scripting](http://forums.adobe.com/community/incopy/incopy_scripting).

For late-breaking InCopy scripting news, see the latest version of this file on the InDesign developer documentation page.

## InCopy Scripting Documentation

The InCopy scripting documentation can be downloaded from the InDesign developer documentation page (mentioned above). The scripting documentation set for InCopy comprises the following:

- *Adobe InCopy Scripting Guide* (AppleScript, JavaScript, and VBScript versions) — Includes a basic tutorial on InCopy scripting, then discusses advanced InCopy scripting topics. All tutorial scripts shown are included in a single ZIP archive, so there is no need to copy and paste scripts from the PDF. (Most scripts shown in the text are incomplete fragments demonstrating a specific property, method, or technique.)
- *JavaScript Tools and Features* — Covers using the ExtendScript Toolkit for JavaScript development, creating user interfaces with ScriptUI, using the File and Folder objects, and other features specific to the ExtendScript language (Adobe’s version of JavaScript).

There is no *Scripting Reference* PDF for this release. Instead, use the object-model viewer included with your script-editing application (as described in *Adobe InCopy Scripting Guide*).

InCopy sample scripts are installed by default, and appear in the Scripts panel (Window > Utilities > Scripts).

### Installing the scripting documentation scripts

In addition to the sample scripts, all scripts shown (in fragmentary form) in the scripting documentation are available for download from the InDesign developer documentation page.

After you download the script ZIP archive and extract the scripts from the archive, move the folder(s) for the language(s) that you want to work with (AppleScript, JavaScript, or VBScript) to your Scripts Panel folder. (For more on installing scripts, see the *Adobe InCopy Scripting Guide*.)

## Running Scripts

To run a script, double-click the script name in the Scripts panel.

On Windows, you can run VBScripts (file extension `.vbs`) or JavaScripts (file extension `.jsx`). On Mac OS, you can run AppleScripts (file extension `.applescript`) or JavaScripts.

## Sample Scripts

Make sure you save your work before running a sample script for the first time.

Before using sample scripts on important InCopy documents, experiment with them so you understand what they do.

InCopy includes the following sample scripts.

Script name	Description	For more information, see page ...
<a href="#">CreateCharacterStyle</a>	Defines a complete character style based on the selected text.	<a href="#">2</a>
<a href="#">ExportAllStories</a>	Exports all stories in a document to a series of text files.	<a href="#">2</a>
<a href="#">FindChangeByList</a>	Performs a series of common text find/change operations by reading a tab-delimited text file.	<a href="#">3</a>
<a href="#">SortParagraphs</a>	Sorts the paragraphs in a selection alphabetically.	<a href="#">3</a>
<a href="#">TabUtilities</a>	Applies tab stops and indents to the selected text.	<a href="#">3</a>

## Script Descriptions

This section includes a brief description of each sample script.

### CreateCharacterStyle

Defines a complete character style based on the selected text.

Demonstrates:

- Processing objects in the selection.
- Creating a character style.
- Getting text-formatting attributes from a text object.
- Filling in character-style properties.

### ExportAllStories

Exports all stories in a document to a series of text files.

Demonstrates:

- Exporting text.
- JavaScript file/folder objects and methods (JavaScript only).
- Creating file names based on ID attributes.
- Creating a user interface.

## FindChangeByList

Performs a series of common text find/change operations by reading a tab-delimited text file.

For more information, see the start of the script or the start of the `FindChangeList.txt` file. You can add your own find/change operations to the `FindChangeList.txt` file or create your own file.

Demonstrates:

- Working with files and folders.
- Finding a file/folder relative to the active script.
- Using the text find/change methods (text, grep, and glyph).
- Reading tab-delimited text data from a text file.

## SortParagraphs

Sorts the paragraphs in a selection alphabetically.

Demonstrates:

- Text-object move method.
- Simple bubble sort.

## TabUtilities

Applies tab stops and indents to the selected text.

Demonstrates:

- Working with tab stops and indents.
- Getting page positions from text objects.
- Getting the text column containing the cursor.
- Creating a user interface.

# Known Issues Related to InCopy Scripting

## Location of JavaScript start-up scripts

User start-up scripts should be placed in the InCopy `start-up scripts` location (where they will run once each time the application is launched), not in the ExtendScript engine initialization scripts location (where they will run each time an engine is initialized).

To run scripts when InCopy starts, put them in the `startup scripts` folder inside the `Scripts` folder in your InCopy folder. (If this folder does not already exist, create it.)

## Scripts run outside InCopy cannot create persistent ExtendScript engines (JavaScript only)

As discussed in Chapter 2, “Scripting,” of *Adobe InCopy Scripting Guide: JavaScript*, ExtendScript scripts can create persistent instances of the ExtendScript engine. Functions and variables defined in the persistent engine can be used by other scripts that execute in that engine. To create a persistent ExtendScript engine, however, the script must be run from the InCopy Scripts panel; running the script from the ExtendScript Toolkit or via BridgeTalk from another application will not create the persistent engine.

## Event listeners added or removed during event propagation are not handled according to the W3C specification

The *W3C Document Object Model (DOM) Level 2 Events Specification* (see <http://www.w3.org/TR/DOM-Level-2-Events/Overview.html>) states the following:

“If an EventListener is added to an EventTarget while it is processing an event, it will not be triggered by the current actions but may be triggered during a later stage of event flow, such as the bubbling phase.”

And:

“If an EventListener is removed from an EventTarget while it is processing an event, it will not be triggered by the current actions. EventListeners can never be invoked after being removed.”

In InCopy scripting, event listeners added to an event target during event propagation are not triggered for the duration of the event. Event listeners removed from an event target during event propagation are still triggered by the event (that is, the event listeners are removed when event processing is complete).

## The ExtendScript toolkit does not show a list of InCopy scripts (Mac OS only)

By default, the ExtendScript Toolkit does not specify a target application when it starts. This means the list of available scripts in the Scripts panel (in the ExtendScript Toolkit, not in InCopy) is not populated with available InCopy scripts. Set the target application to InCopy, and the ExtendScript Toolkit populates the Scripts panel.