Issue: Options to handle the tab/browser close scenario when there is a delay rendered format written by the web author (https://github.com/w3c/editing/issues/424#issuecomment-1506128779)

Option 1

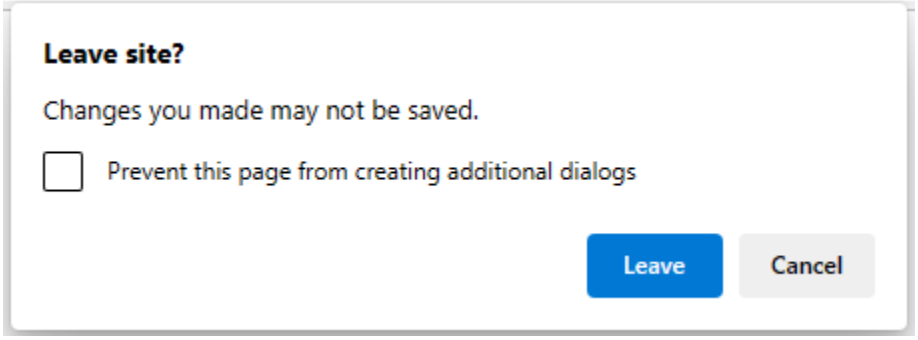
Add a `beforeunload` event listener: A web author could choose to `preventDefault beforeunload` event and notify the user that there is some data pending to be written into the clipboard. In this event handler, web authors could use a new clipboard API (like `navigator.clipboard.undelayFormats([""])` or something similar) to write the data for the formats that were delay rendered.

e.g.

```
function generateExpensiveCustomFormatBlob() {
// Produce the expensive data here...
....
return new Blob....
}
const clipboard_item = new ClipboardItem({
                'text/html': Promise.resolve(generateExpensiveCustomFormatBlob)
                });
navigator.clipboard.write([clipboard_item]);

const beforeUnloadListener = (event) => {
      event.preventDefault();
       navigator.clipboard.undelayFormats(['text/html']);
      return (event.returnValue = "");
 };
```

When the user closes the tab/browser, beforeunload event shows a dialog that there are some changes in the page that may get lost, and the user has the option to either leave the page and discard the changes, or let the site write all the delay rendered formats to the clipboard.
The text in the dialog is very generic. It's the same text we see when the user partially fills out a form and closes the tab/browser or the web authors preventDefault beforeunload event.



If the user clicks on “Cancel”, then we can let the site produce the data for the delay rendered formats. If the user clicks on “Leave”, then we cancel all the delay rendered formats callbacks and either empty the clipboard, or write a subset of the formats that we have the data for and clear the rest. Writing a subset of formats during beforeunload event would affect the navigation time, so we could introduce a timer to mitigate the delay. Emptying the clipboard leads to loss of data, and providing only a subset of the formats leads to loss of fidelity. In the native Excel app, the latter approach was chosen so maybe we could pursue something like this that wouldn’t lead to loss of data for the user which may impact the user experience on the platform.
Problem: ~~Registering a beforeunload event handler would disable BFCache.~~ It affects navigation/close time and performance of the page when the user performs back and forward navigations.

Edit: Our assumption that BFCache would be disabled in Chromium if a beforeunload event handler was registered, is incorrect (per Fergal Daly’s comment here).

Option 2

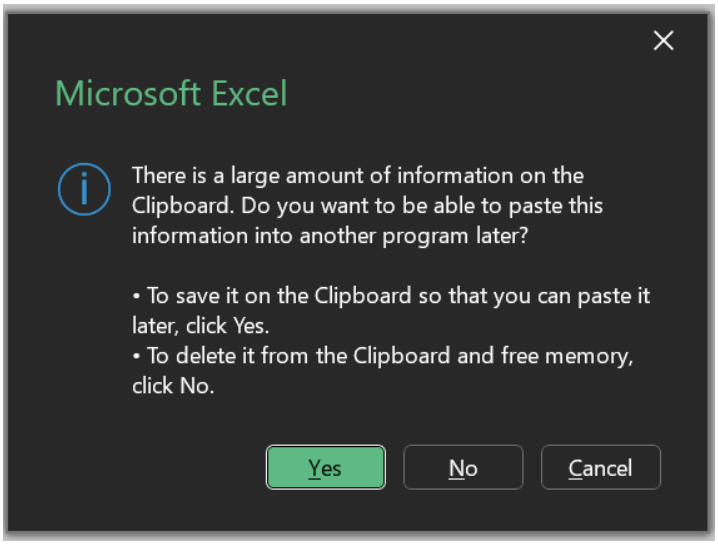
Browser could choose to trigger all the callbacks before it fires `beforeunload` event so it could populate the data for the delay rendered formats. ~~That way the web authors don't have to register for `beforeunload` event which would have disabled bfcache.~~ In this case, a timeout can also be added to prevent sites from abusing the delay rendering of formats.

User clicks on the tab/browser close that triggers all the callbacks for the delay rendered formats. Once the data for all the formats are written into the clipboard, beforeunload event is fired and then shutdown is triggered.
This slows down the tab/browser close process. For a few seconds, the browser/tab would just freeze because the clipboard write is in-progress as it happens in the browser process. Generation of data in the renderer process for all delay rendered formats also adds to the time it takes for shutdown. This indirectly affects navigation as well since the user could choose to press the forward/back button and paste the copied content in a different page.

Option 3

Browsers could choose to show a confirmation dialog if there are any delay rendered formats in the clipboard: Although it is similar to option 1, here the text of the dialog explicitly mentions delay rendering of formats which is more informative to the user. This option has similar concerns as option 1 except the bfcache part. BFCache is not directly impacted by this as there is no beforeunload event handler registered, but it does affect navigation time since it takes time for the site to produce the content for all the formats, so that slows down browser/tab shutdown behavior.

When user clicks on the tab/browser close button, sites could show a dialog like below:



This is what native Excel app does, but UAs can make the dialog generic enough for all sites that support delayed rendering of formats.

Option 4

Only allow a subset of formats to be delay rendered, and force the web authors to produce one inexpensive format (such as plain text). Throw away all the delay rendered formats and put empty data for those formats in the clipboard when the tab/browser is closed.

If a user tries to paste content in MSPaint and there is no image format on the clipboard, then the paste would fail silently. To the user it would appear as if nothing was copied. This “loss of data” would have a negative effect on the user’s copy/paste experience in the browser.

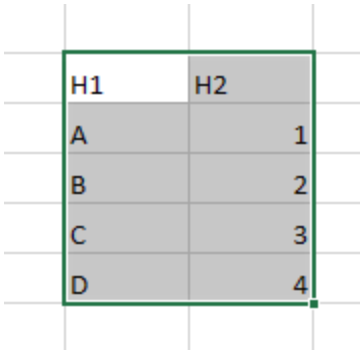
In apps that support formats that have high fidelity content, the pasted content would be of lower fidelity as the delayed formats are empty. For a user, if they are able to copy-paste content without the rich formatting, it may not be a bad experience. e.g. copying from Excel and pasting it in the Notepad would paste only the plain text representation of the Excel spreadsheet content. But, in apps like Excel, Word etc where the formatting matters, it could be a bad user experience if we paste lower fidelity content.

e.g.

User copies the below content from native Excel:

H1	H2
A	1
B	2
C	3
D	4

Pastes it in Excel online and only the plain text format is present on the clipboard:



H1	H2
A	1
B	2
C	3
D	4