

DOCX Download Button Issue - Complete Analysis

Problem Summary

The "Download DOCX" button in the resume preview window doesn't do anything when clicked.

Root Causes Identified

1. Cross-Window Function Access Issue

Location: `script.js` - `openPrintableResume()` function

Current Code:

```
javascript

<button onclick="window.opener.generateDocxFromHTML(window.opener.resumeData, document)"
    class="btn btn-primary"
    style="background-color: #2b579a;">
    Download DOCX
</button>
```

Problem: The button tries to call `generateDocxFromHTML` from the parent window (`window.opener`), but there are scoping and timing issues with this approach.

2. Library Availability

Location: `index.html` - `<head>` section

Current Code:

```
html

<script src="https://cdnjs.cloudflare.com/ajax/libs/html-docx-js/0.4.0/html-docx.js"></script>
```

Problem: The `html-docx-js` library is loaded in the main window, but the print window (opened via `window.open()`) is a completely separate document that doesn't inherit script references from the parent.

3. Function Exposure Timing

Location: `script.js` - End of `openPrintableResume()` function

Current Code:

```
javascript

printWindow.document.write(html);
printWindow.document.close();

// Expose generateDocxFromHTML to the new window
printWindow.generateDocxFromHTML = generateDocxFromHTML;
```

Problem: This assignment happens after `document.write()` completes, but the HTML with the button is already rendered, creating a potential race condition.

Recommended Solution

Option 1: Parent Window Handles Everything (Simplest)

This approach keeps all the logic in the parent window where the library is already loaded.

Step 1: Update the Button in `openPrintableResume()`

Find this in `script.js` around line 420:

```
javascript

<button onclick="window.opener.generateDocxFromHTML(window.opener.resumeData, document)"
    class="btn btn-primary"
    style="background-color: #2b579a;">
    Download DOCX
</button>
```

Replace with:

```
javascript

<button onclick="window.opener.downloadResumeAsDocx()"
    class="btn btn-primary"
    style="background-color: #2b579a;">
    Download DOCX
</button>
```

Step 2: Add Helper Function to `script.js`

Add this function anywhere in `script.js` (suggested: after `generateDocxFromHTML()`):

```
javascript

function downloadResumeAsDocx() {
    // This function is called from the print window
    // but executes in the parent window context where the library is available
    generateDocxFromHTML(resumeData, null);
}
```

Step 3: Update `generateDocxFromHTML()` to Handle Null Document

Find the `(generateDocxFromHTML())` function and update the beginning:

```
javascript

function generateDocxFromHTML(data, targetDoc) {
    // Get the template HTML
    let resumeContent;
    if (data.selectedTemplate === 'classic') {
        resumeContent = getClassicTemplateHTML(data, true); // Pass true for DOCX mode
    } else if (data.selectedTemplate === 'executive') {
        resumeContent = getExecutiveTemplateHTML(data);
    } else {
        resumeContent = getModernTemplateHTML(data);
    }

    // Get CSS
    const css = getResumeCSS();

    // ... rest of function
}
```

Option 2: Load Library in Print Window (More Complex)

This approach loads the library in the print window itself.

Step 1: Update the HTML in `(openPrintableResume())`

Find the `<head>` section in the HTML template string and add:

```
javascript

const html = `

<!DOCTYPE html>
<html>
<head>
  <title>${data.contact.fullName} - Resume</title>
  <base href="${baseUrl}">
  <link rel="stylesheet" href="css/styles.css?v=${new Date().getTime()}">

  <!-- Add DOCX library to print window -->
  <script src="https://cdnjs.cloudflare.com/ajax/libs/html-docx/0.4.0/html-docx.js"></script>

  <style>
    <!-- ... existing styles ... -->
  </style>
</head>
<!-- ... rest ... -->
`;
```

Step 2: Update Button

```
javascript

<button onclick="downloadDocx()"
  class="btn btn-primary"
  style="background-color: #2b579a;">
  Download DOCX
</button>
```

Step 3: Add Script in Print Window

In the same HTML template, add this to the bottom `<script>` tag:

```
javascript
```

```
<script>

window.onload = function() {
    // ... existing code ...
};

function downloadDocx() {
    const data = window.opener.resumeData;

    // Copy the function logic here or reference parent
    let resumeContent;
    if (data.selectedTemplate === 'classic') {
        resumeContent = window.opener.getClassicTemplateHTML(data, true);
    } else if (data.selectedTemplate === 'executive') {
        resumeContent = window.opener.getExecutiveTemplateHTML(data);
    } else {
        resumeContent = window.opener.getModernTemplateHTML(data);
    }

    const css = window.opener.getResumeCSS();
    const resumeHTML = `

        <!DOCTYPE html>
        <html>
        <head>
            <meta charset="UTF-8">
            <style>${css}</style>
        </head>
        <body>${resumeContent}</body>
    </html>
`;

    const converted = htmlDocx.asBlob(resumeHTML, {
        orientation: 'portrait',
        margins: { top: 720, right: 720, bottom: 720, left: 720 }
    });

    const url = URL.createObjectURL(converted);
    const a = document.createElement('a');
    a.href = url;
    const filename = (data.contact && data.contact.fullName) ?
        `${data.contact.fullName.replace(/\s+/g, '_')}_Resume.docx` :
        'Resume.docx';
    a.download = filename;
    document.body.appendChild(a);
}
```

```
a.click();

setTimeout(() => {
    URL.revokeObjectURL(url);
    document.body.removeChild(a);
}, 100);
}

</script>
```

Recommendation

Use **Option 1 (Parent Window Handles Everything)** because:

1. Simpler implementation
 2. Library already loaded in parent window
 3. Fewer points of failure
 4. Less code duplication
 5. Easier to maintain
-

Testing Steps

After implementing the fix:

1. Open the resume builder
 2. Fill in some information
 3. Click "Create Resume"
 4. In the preview window, click "Download DOCX"
 5. Verify that a `.docx` file downloads
 6. Open the file in Word/LibreOffice to confirm formatting
-

Additional Notes

Browser Console Debugging

If the button still doesn't work after the fix, check the browser console (F12) for errors:

- Look for `ReferenceError` messages
- Check for CORS issues with the CDN library
- Verify `window.opener` is accessible (it won't work if the user has strict privacy settings)

Alternative Library

If `html-docx-js` continues to cause issues, consider:

- **docx.js** - More modern, better formatting
 - **docx** package - More control, requires build step
 - Server-side generation using Python (`python-docx`) or Node.js
-

Quick Fix Summary

Minimal changes required:

1. In `script.js`, find line ~420 and change button onclick to:

```
javascript
onclick="window.opener.downloadResumeAsDocx()"
```

2. In `script.js`, add after `generateDocxFromHTML()`:

```
javascript
function downloadResumeAsDocx() {
  generateDocxFromHTML(resumeData, null);
}
```

3. Test the download functionality

That's it! The button should now work properly.