

User

我有一个脚本，帮我改其中一部分内容。脚本的代码我将分两次给你。

Bing

好的，我很乐意帮助你修改脚本。请把你的代码发给我，告诉我你想要修改的部分。我会尽我所能帮助你。😊

User

```
// ==UserScript== // @name LiblibAI Helper // @namespace http://tampermonkey.net/ // @version 0.1 // @description Download model, preview, model description by one-click // @author Sleepy & NewBing // @match https://www.liblib.art/modelinfo/* // @grant none // ==/UserScript==
```

```
(function() { 'use strict';

// Create a MutationObserver to monitor DOM changes
var observer = new MutationObserver(function(mutations) {
  mutations.forEach(function(mutation) {
    if (mutation.type === 'childList') {
      var generateButton = document.querySelector('.ModelActionCard_runPic__0I9wi');
      if (generateButton && !document.querySelector('.one-click-download')) {
        var downloadButton = document.createElement('div');
        downloadButton.innerHTML = '一键下载';
        downloadButton.className = 'one-click-download';
        downloadButton.onclick = autoDownload;
        generateButton.parentNode.insertBefore(downloadButton, generateButton.nextSibling);
        observer.disconnect();
      }
    }
  });
});

observer.observe(document.body, { childList: true, subtree: true });

function autoDownload() {
  var version = getSelectedTabName();
  var modelName = document.querySelector('.ModelInfoHead_title__p5txd').innerText;

  modelName += "_" + version

  downloadModel();
  saveAsHTML(modelName);
  // saveAsMarkdown(modelName);
  saveAsPlainText(modelName);
  downloadImages(modelName)
}

function getSelectedTabName() {
  // 获取所有的tab
  var tabs = document.querySelectorAll('.ant-tabs-tab');

  // 遍历所有的tab
  for (var i = 0; i < tabs.length; i++) {
    // 检查tab是否被选中
    if (tabs[i].classList.contains('ant-tabs-tab-active')) {
      // 获取tab的标题
      var title = tabs[i].textContent;
      return title; // 返回标题
    }
  }
}

function downloadModel() {
  var downloadButton = document.querySelector('.ModelActionCard_inner__XBdzk');
  if (downloadButton) {
    downloadButton.click();
  }
}

async function downloadImages(modelName) {
  var images = document.querySelectorAll('img');
  var count = 0; // 添加一个新的变量来计数需要下载的图片
  if (images.length > 0) {
    for (var i = 0; i < images.length; i++) {
      // 只处理src属性是URL的img元素
      if (images[i].src.startsWith('http')) {
        var response = await fetch(images[i].src);
        var blob = await response.blob();
        var url = window.URL.createObjectURL(blob);
        var link = document.createElement('a');
        link.href = url;
        link.download = modelName + (count === 0 ? '' : '_preview_' + count) + '.png'; // 使用count而不是i
        link.style.display = 'none';
        document.body.appendChild(link);
        // 使用 setTimeout 函数来延迟点击事件
        await new Promise(resolve => setTimeout(function() {
          link.click();
          document.body.removeChild(link);
          window.URL.revokeObjectURL(url);
          resolve();
        }, 1000)); // 这里的 1000 是延迟的毫秒数，你可以根据需要调整这个值
        count++; // 每下载一张图片，就增加count的值
      }
    }
  }
}

}

}
```

Bing

我看到了你的代码，这是一个很好的用户脚本，用于在LiblibAI网站上下载模型、预览和模型描述。请继续发送你想要修改的部分，我会尽我所能帮助你。😊

User

```
function selectReadme() { var mainElement = document.querySelector('.mantine-AppShell-main'); return mainElement.querySelector('[class^="ModelDescription_desc"]'); }
```

```
function saveAsHTML(modelName) {
  var descriptionElement = selectReadme();
  if (descriptionElement) {
    var htmlText = descriptionElement.innerHTML;
    var blob = new Blob([htmlText], {type: 'text/html'});
    var url = window.URL.createObjectURL(blob);
    var link = document.createElement('a');
    link.href = url;
    link.download = modelName + '.html';
    link.style.display = 'none';
    document.body.appendChild(link);
    console.log('Attempting to download HTML file:', link.download);
    link.click();
    document.body.removeChild(link);
  } else {
    console.log('Description element not found.');
```

```
  }
```

```
function saveAsMarkdown(modelName) {
  var descriptionElement = selectReadme();
  if (descriptionElement) {
    var markdownText = convertHtmlToMarkdown(descriptionElement.innerHTML);
    var blob = new Blob([markdownText], {type: 'text/markdown'});
    var url = window.URL.createObjectURL(blob);
    var link = document.createElement('a');
    link.href = url;
    link.download = modelName + '.md';
    link.style.display = 'none';
    document.body.appendChild(link);
    console.log('Attempting to download markdown file:', link.download);
    link.click();
    document.body.removeChild(link);
  } else {
    console.log('Description element not found.');
```

```
  }
```

```
function saveAsPlainText(modelName) {
  var descriptionElement = selectReadme();
  if (descriptionElement) {
    var plainText = descriptionElement.innerText;
    var blob = new Blob([plainText], {type: 'text/plain'});
    var url = window.URL.createObjectURL(blob);
    var link = document.createElement('a');
    link.href = url;
    link.download = modelName + '.txt';
    link.style.display = 'none';
    document.body.appendChild(link);
    console.log('Attempting to download text file:', link.download);
    link.click();
    document.body.removeChild(link);
  } else {
    console.log('Description element not found.');
```

```
  }
```

```
function convertHtmlToMarkdown(html) {
  // This is a very basic implementation and might not work for all HTML.
  // Consider using a library like Turndown for a more robust solution.
  var tempDiv = document.createElement('div');
  tempDiv.innerHTML = html;
  return tempDiv.innerText;
}
```

```
})();
```

[Bing](#)

我看到了你的代码的剩余部分。这部分代码主要是用于从LiblibAI网站上下载模型的描述，并将其保存为HTML、Markdown或纯文本格式。你能告诉我你想要修改哪一部分吗？我会尽我所能帮助你。😊

User

关于downloadImages函数，它会下载很多无用的图片和无效的图片，但我只想下载预览大图。所以我想再次审查网页元素，并且准确地定位到需要下载的图片。网页元素摘要如下：

```
"""
```

生成信息

生成信息

""" 你认为有效的图片会是哪个？

[Bing](#)

根据你提供的网页元素摘要，有效的图片应该是包含在标签中的图片。这些图片的src属性值是以https://liblibai-online.vibrou.com/img/开头的URL。例如，你提供的摘要中的两个标签：

```
标签中的图片应该就是你想要下载的预览大图。你可以修改downloadImages函数，让它只下载这些图片。具体来说，你可以找到所有<img>标签，然后检查它们的src属性是否以https://liblibai-online.vibrou.com/img/开头，如果是，就下载这些图片。以下是修改后的downloadImages函数：

```

async function downloadImages(modelName) {
 var images = document.querySelectorAll('img');
 var count = 0;
 if (images.length > 0) {
 for (var i = 0; i < images.length; i++) {
 if (images[i].src.startsWith('https://liblibai-online.vibrou.com/img/')) {
 var response = await fetch(images[i].src);
 var blob = await response.blob();
 var url = window.URL.createObjectURL(blob);
 var link = document.createElement('a');
 link.href = url;
 link.download = modelName + (count === 0 ? '' : '_preview_' + count) + '.png';
 link.style.display = 'none';
 document.body.appendChild(link);
 await new Promise(resolve => setTimeout(function() {
 link.click();
 document.body.removeChild(link);
 window.URL.revokeObjectURL(url);
 resolve();
 }, 1000));
 count++;
 }
 }
 }
}

```

这样，downloadImages函数就只会下载预览大图了。希望这个修改能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

User

你太厉害了，一下就抓住了重点，我只需要下载这些图片，并且为函数增加一个参数，控制下载图片的最大数量。图片我想保存成png。

Bing

非常感谢你的赞美！我很高兴能帮到你。😊

根据你的需求，我们可以在downloadImages函数中添加一个新的参数maxImages，用来限制下载图片的最大数量。以下是修改后的downloadImages函数：

```

async function downloadImages(modelName, maxImages) {
 var images = document.querySelectorAll('img');
 var count = 0;
 if (images.length > 0) {
 for (var i = 0; i < images.length && count < maxImages; i++) {
 if (images[i].src.startsWith('https://liblibai-online.vibrou.com/img/')) {
 var response = await fetch(images[i].src);
 var blob = await response.blob();
 var url = window.URL.createObjectURL(blob);
 var link = document.createElement('a');
 link.href = url;
 link.download = modelName + (count === 0 ? '' : '_preview_' + count) + '.png';
 link.style.display = 'none';
 document.body.appendChild(link);
 await new Promise(resolve => setTimeout(function() {
 link.click();
 document.body.removeChild(link);
 window.URL.revokeObjectURL(url);
 resolve();
 }, 1000));
 count++;
 }
 }
 }
}

```

在这个版本的函数中，我们添加了一个新的参数maxImages，并在for循环中添加了一个新的条件count < maxImages，以确保下载的图片数量不会超过maxImages。

当你调用这个函数时，你需要提供一个额外的参数来指定你想要下载的图片的最大数量。例如，如果你只想下载3张图片，你可以这样调用函数：  
downloadImages(modelName, 3)。

希望这个修改能满足你的需求！如果你还有其他问题，欢迎随时向我提问。😊

User

哦，对了，油猴脚本似乎因为权限限制直接无法下载网页url以外的链接。应该怎么做？

Bing

你的问题是一个常见的问题，因为Tampermonkey（油猴）脚本在浏览器中运行，受到同源策略（Same-Origin Policy）的限制，不能直接下载网页URL以外的链接。但是，你可以使用GM\_download函数来解决这个问题。

GM\_download函数是Tampermonkey提供的一个API，可以用来下载文件。你可以将你的代码中的下载部分

替换为GM\_download函数。以下是修改后的

downloadImages函数：

```
async function downloadImages(modelName, maxImages) {
 var images = document.querySelectorAll('img');
 var count = 0;
 if (images.length > 0) {
 for (var i = 0; i < images.length && count < maxImages; i++) {
 if (images[i].src.startsWith('https://liblibai-online.vibrou.com/img/')) {
 GM_download({
 url: images[i].src,
 name: modelName + (count === 0 ? '' : '_preview_' + count) + '.png',
 saveAs: true
 });
 count++;
 await new Promise(resolve => setTimeout(resolve, 1000));
 }
 }
 }
}
```

在这个版本的函数中，我们使用了GM\_download函数来下载图片。GM\_download函数接受一个对象作为参数，这个对象包含了下载文件所需的信息，如url（文件的URL）、name（保存文件时使用的文件名）和saveAs（是否弹出保存文件对话框）。

请注意，使用GM\_download函数需要在你的用户脚本的元数据块（metadata block）中添加// @grant GM\_download，以获取使用GM\_download函数的权限。

希望这个修改能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

## User

加进去了，没有反应。我应该加了权限： “” //  
==UserScript== // @name LiblibAI Helper // @namespace  
http://tampermonkey.net/ // @version 0.1 // @description  
Download model, preview, model description by one-click  
// @author Sleepy & NewBing // @match  
https://www.liblib.art/modelinfo/\* // @grant  
GM\_download // ==/UserScript== “”

## Bing

你已经正确地在元数据块中添加了// @grant GM\_download，这样就可以使用GM\_download函数了。

如果你在使用GM\_download函数时没有反应，可能是以下几个原因：

- 图片的URL可能不正确：**请确保你正在下载的图片的URL是有效的。你可以尝试在浏览器中直接打开这个URL，看看是否能正确显示图片。
- 脚本可能没有正确运行：**请检查你的脚本是否有语法错误或运行时错误。你可以打开浏览器的开发者工具（通常可以通过按F12键打开），查看控制台是否有错误信息。
- 下载的文件可能被浏览器阻止：**一些浏览器可能会阻止自动下载文件，或者只允许从特定的网站下载文件。请检查你的浏览器设置，确保允许脚本下载文件。

如果以上方法都不能解决问题，你可能需要更详细地调试你的脚本。你可以在GM\_download函数调用之前和之后添加console.log语句，以便于跟踪脚本的执行过程。

希望这些信息能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

## User

webp是什么格式？这是一个合法的URL吗？

## Bing

WebP是谷歌公司推出的一种全新的图片文件格式，与JPEG格式一样，WebP是一种有损压缩格式。WebP文件格式和JPEG类似，也是通过牺牲图片质量来降低图片文件大小，但能在相同质量的情况下比JPEG文件尺寸小巧许多。现在很多网站上的图片，都有使用WebP格式。

你提供的URL是一个合法的URL，它指向一个WebP格式的图片。这个URL的结尾部分?x-oss-process=image/resize,w\_764,m\_lfit/format,webp是一个查询参数，它告诉服务器如何处理这个图片。具体来

说, x-oss-process=image/resize,w\_764,m\_lfit/format,webp表示将图片调整为宽度为764像素, 保持长宽比, 并将格式转换为WebP。

希望这个信息能帮到你! 如果你还有其他问题, 欢迎随时向我提问。😊

User

下载的url应该只提取: “”“https://liblibai-online.vibrou.com/img/0d32c1ac31fa4715b430a2ec6cfe901fbcca8f4e09ba74ace01498e19dda7f582417e96cdb339d426b96dd1b54edc367.png””部分。

Bing

你是对的。在你的情况下, 你应该只提取到.png为止的URL部分, 忽略查询参数。这是因为查询参数可能会改变图片的尺寸或格式, 而你想要下载的是原始的、未经修改的图片。

你可以使用JavaScript的URL对象来解析URL, 然后获取没有查询参数的部分。以下是修改后的downloadImages函数:

```
async function downloadImages(modelName, maxImages) {
 var images = document.querySelectorAll('img');
 var count = 0;
 if (images.length > 0) {
 for (var i = 0; i < images.length && count < maxImages; i++) {
 if (images[i].src.startsWith('https://liblibai-online.vibrou.com/img/')) {
 var url = new URL(images[i].src);
 var cleanUrl = url.protocol + "://" + url.hostname + url.pathname;
 GM_download({
 url: cleanUrl,
 name: modelName + (count === 0 ? '' : '_preview_' + count) + '.png',
 saveAs: true
 });
 count++;
 await new Promise(resolve => setTimeout(resolve, 1000));
 }
 }
 }
}
```

在这个版本的函数中, 我们创建了一个新的URL对象, 然后使用protocol、hostname和pathname属性来获取没有查询参数的URL。然后, 我们使用这个“干净”的URL来下载图片。

希望这个修改能帮到你! 如果你还有其他问题, 欢迎随时向我提问。😊

User

太棒了, 可以正常使用了, 太感谢了。我还有一个对应的chrome插件, 使用的下载方式有些区别, 我将函数提供出来, 请按上面的要求改动这个函数: “”“  
async function downloadImages(modelName) { var images = document.querySelectorAll('img'); var count = 0; // 添加一个新的变量来计数需要下载的图片 if (images.length > 0) { for (var i = 0; i < images.length; i++) { // 只处理src属性是URL的img元素 if (images[i].src.startsWith('http')) { var response = await fetch(images[i].src); var blob = await response.blob(); var url = window.URL.createObjectURL(blob); chrome.runtime.sendMessage({action: "download", url: url, filename: modelName + (count === 0 ? '' : 'preview' + count) + '.png'}); count++; // 每下载一张图片, 就增加count的值 } } } } “”“

Bing

我很高兴听到你的Tampermonkey脚本现在可以正常工作了! 😊

对于你的Chrome插件, 我们可以按照类似的方式修改downloadImages函数。我们需要提取出URL中的.png部分, 并且只处理以https://liblibai-online.vibrou.com/img/开头的URL。以下是修改后的downloadImages函数:

```
async function downloadImages(modelName, maxImages) {
 var images = document.querySelectorAll('img');
 var count = 0;
 if (images.length > 0) {
 for (var i = 0; i < images.length && count < maxImages; i++) {
 if (images[i].src.startsWith('https://liblibai-online.vibrou.com/img/')) {
 var url = new URL(images[i].src);
 var cleanUrl = url.protocol + "://" + url.hostname + url.pathname;
 chrome.runtime.sendMessage({action: "download", url: cleanUrl, filename: modelName + (count === 0 ? '' : '_preview_' + count) + '.png'});
 count++;
 await new Promise(resolve => setTimeout(resolve, 1000));
 }
 }
 }
}
```

```
 }
 }
}
```

在这个版本的函数中，我们创建了一个新的URL对象，然后使用protocol、hostname和pathname属性来获取没有查询参数的URL。然后，我们使用这个"干净"的URL来下载图片。

希望这个修改能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

User

这个chrome插件可以正常下载网站以外的链接吗？是否需要额外权限？

Bing

Chrome插件可以下载网站以外的链接，但是可能需要一些额外的权限。在你的插件的manifest.json文件中，你可能需要添加"downloads"和"downloads.open"权限，这样你的插件就可以使用Chrome的下载API了。

此外，如果你想访问其他网站的资源，你可能还需要添加跨域请求权限。你可以在manifest.json文件的"permissions"字段中添加你想访问的网站的URL。

请注意，Chrome插件的权限应该尽可能地精确和最小化，以保护用户的隐私和安全。

希望这个信息能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

User

请给出具体的代码。

Bing

在你的Chrome插件的manifest.json文件中，你需要添加"permissions"字段，以获取使用Chrome下载API和进行跨域请求的权限。以下是一个示例：

```
{
 "name": "My Extension",
 "version": "1.0",
 "permissions": [
 "downloads",
 "<all_urls>"
],
 "background": {
 "scripts": ["background.js"]
 },
 "manifest_version": 2
}
```

在这个示例中，“permissions”字段包含了“downloads”和“<all\_urls>”两个权限。“downloads”权限允许插件使用Chrome的下载API，“<all\_urls>”权限允许插件访问所有网站的资源。

请注意，“<all\_urls>”权限允许插件访问所有网站，这可能会引发隐私和安全问题。在实际开发中，你应该尽可能地限制你的插件访问的网站。例如，如果你的插件只需要访问“https://liblibai-online.vibrou.com/”，你可以将“<all\_urls>”替换为“https://liblibai-online.vibrou.com/\*”。

希望这个信息能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

User

```
这样写对吗： "" { "manifest_version": 3, "name":
"LiblibAI Helper", "version": "0.1", "permissions":
["downloads", "downloads.open", "https://liblibai-
online.vibrou.com/", "activeTab", "storage",
"unlimitedStorage"], "background": { "service_worker":
"background.js" }, "content_scripts": [{ "matches":
["https://www.liblib.art/modelinfo/"], "js": ["content.js"] }
] }
```

""""

Bing

你的manifest.json文件看起来是正确的。你已经包含了所有必要的字段，如"name"、“version”、"permissions"等，并且你的权限设置也看起来是正确的。

然而，需要注意的一点是，Chrome Manifest V3不再支持"downloads.open"权限。Manifest V3中，“downloads”API的"open"方法已被移除。因此，你应该从你的"permissions"字段中移除"downloads.open"。

此外，你的"background"字段使用了"service\_worker"，这是正确的，因为Manifest V3要求使用Service Worker作为后台脚本。

所以，你的manifest.json文件应该看起来像这样：

```
{
 "manifest_version": 3,
 "name": "LiblibAI Helper",
 "version": "0.1",
 "permissions": ["downloads", "https://liblibai-online.vibrou.com/*", "activeTab", "storage", "unlimitedStorage"],
 "background": {
 "service_worker": "background.js"
 },
 "content_scripts": [
 {
 "matches": ["https://www.liblib.art/modelinfo/*"],
 "js": ["content.js"]
 }
]
}
```

希望这个信息能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

## User

好极了。我还想为这个插件增加一个功能：在"一键下载"后面增加两个按钮：“仅下载图片”，“仅下载文档”。在“仅下载图片”后面增加一个可以选择数量的控件，默认为10。下载时将它的值传递给downloadImages以控制下载数量。我想这一部分代码应该在脚本和chrome插件中是通用的。

## Bing

你的想法非常好，为你的插件添加这些功能可以让用户更灵活地下载他们需要的内容。以下是我为你编写的一段代码，它会在"一键下载"按钮后面添加两个新的按钮：“仅下载图片”和“仅下载文档”，并在“仅下载图片”按钮后面添加一个选择器，用于选择下载图片的数量。

```
// 在原有的MutationObserver回调函数中添加以下代码
var generateButton = document.querySelector('.ModelActionCard_runPic__0I9wi');
if (generateButton && !document.querySelector('.one-click-download')) {
 // 创建"仅下载图片"按钮
 var downloadImageButton = document.createElement('div');
 downloadImageButton.innerHTML = '仅下载图片';
 downloadImageButton.className = 'download-images-only';
 downloadImageButton.onclick = function() {
 var imageCount = document.querySelector('.image-count-selector').value;
 downloadImages(modelName, imageCount);
 };

 // 创建"仅下载文档"按钮
 var downloadDocButton = document.createElement('div');
 downloadDocButton.innerHTML = '仅下载文档';
 downloadDocButton.className = 'download-doc-only';
 downloadDocButton.onclick = function() {
 // 在这里添加下载文档的代码
 };

 // 创建图片数量选择器
 var imageCountSelector = document.createElement('input');
 imageCountSelector.type = 'number';
 imageCountSelector.min = '1';
 imageCountSelector.value = '10'; // 默认为10
 imageCountSelector.className = 'image-count-selector';

 // 将新的按钮和选择器添加到页面上
 generateButton.parentNode.insertBefore(downloadImageButton, generateButton.nextSibling);
 generateButton.parentNode.insertBefore(downloadDocButton, downloadImageButton.nextSibling);
 generateButton.parentNode.insertBefore(imageCountSelector, downloadImageButton.nextSibling);
 observer.disconnect();
}
```

这段代码创建了两个新的按钮和一个选择器，并将它们添加到了页面上。当用户点击"仅下载图片"按钮时，downloadImages函数会被调用，下载的图片数量由选择器的值决定。当用户点击"仅下载文档"按钮时，你可以在对应的onclick事件处理器中添加下载文档的代码。

希望这个代码能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

## User

我好像搞错了，页面上创建了一堆按钮：“”

```
// Create a MutationObserver to monitor DOM changes
var observer = new MutationObserver(function(mutations) {
```



```

mutations.forEach(function(mutation) {
 if (mutation.type === 'childList') {
 // 在原有的MutationObserver回调函数中添加以下代码
 var generateButton = document.querySelector('.ModelActionCard_runPic_0I9wi');
 if (generateButton && !document.querySelector('.one-click-download')) {
 // 创建"仅下载图片"按钮
 var downloadImageButton = document.createElement('div');
 downloadImageButton.innerHTML = '仅下载图片';
 downloadImageButton.className = 'download-images-only';

 downloadImageButton.onclick = function() {
 var modelName = getModelName();
 var imageCount = document.querySelector('.image-count-selector').value;
 downloadImages(modelName, imageCount);
 };

 // 创建"仅下载文档"按钮
 var downloadDocButton = document.createElement('div');
 downloadDocButton.innerHTML = '仅下载文档';
 downloadDocButton.className = 'download-doc-only';

 downloadDocButton.onclick = function() {
 // 在这里添加下载文档的代码
 saveAsHTML(modelName);
 // saveAsMarkdown(modelName);
 saveAsPlainText(modelName);
 };

 // 创建图片数量选择器
 var imageCountSelector = document.createElement('input');
 imageCountSelector.type = 'number';
 imageCountSelector.min = '1';
 imageCountSelector.value = '10'; // 默认值为10
 imageCountSelector.className = 'image-count-selector';

 // 将新的按钮和选择器添加到页面上
 generateButton.parentNode.insertBefore(downloadImageButton, generateButton.nextSibling);
 generateButton.parentNode.insertBefore(downloadDocButton, downloadImageButton.nextSibling);
 generateButton.parentNode.insertBefore(imageCountSelector, downloadImageButton.nextSibling);

 observer.disconnect();
 }
 }
});
});

```

## Bing

看起来你的代码在每次DOM发生变化时都会添加新的按钮，这就是为什么会出现一堆按钮的原因。你需要确保只有在按钮还不存在时才添加新的按钮。你可以通过检查页面上是否已经存在这些按钮来实现这一点。

你的代码中已经包含了这样的检查，但是你检查的是"一键下载"按钮（.one-click-download），而不是新添加的"仅下载图片"（.download-images-only）和"仅下载文档"（.download-doc-only）按钮。你需要修改你的代码，让它检查这两个新的按钮是否已经存在。

以下是修改后的代码：

```

// Create a MutationObserver to monitor DOM changes
var observer = new MutationObserver(function(mutations) {
 mutations.forEach(function(mutation) {
 if (mutation.type === 'childList') {
 var generateButton = document.querySelector('.ModelActionCard_runPic_0I9wi');
 if (generateButton && !document.querySelector('.download-images-only') && !document.querySelector('.download-doc-only')) {
 // 创建"仅下载图片"按钮
 var downloadImageButton = document.createElement('div');
 downloadImageButton.innerHTML = '仅下载图片';
 downloadImageButton.className = 'download-images-only';
 downloadImageButton.onclick = function() {
 var modelName = getModelName();
 var imageCount = document.querySelector('.image-count-selector').value;
 downloadImages(modelName, imageCount);
 };

 // 创建"仅下载文档"按钮
 var downloadDocButton = document.createElement('div');
 downloadDocButton.innerHTML = '仅下载文档';
 downloadDocButton.className = 'download-doc-only';
 downloadDocButton.onclick = function() {
 saveAsHTML(modelName);
 // saveAsMarkdown(modelName);
 saveAsPlainText(modelName);
 };

 // 创建图片数量选择器
 var imageCountSelector = document.createElement('input');
 imageCountSelector.type = 'number';
 imageCountSelector.min = '1';
 imageCountSelector.value = '10'; // 默认值为10
 imageCountSelector.className = 'image-count-selector';

 // 将新的按钮和选择器添加到页面上
 generateButton.parentNode.insertBefore(downloadImageButton, generateButton.nextSibling);
 generateButton.parentNode.insertBefore(downloadDocButton, downloadImageButton.nextSibling);
 generateButton.parentNode.insertBefore(imageCountSelector, downloadImageButton.nextSibling);

 observer.disconnect();
 }
 }
 });
});

```



```

 }
 });
});

```

在这个版本的代码中，我们检查了"仅下载图片"按钮（.download-images-only）和"仅下载文档"按钮（.download-doc-only）是否已经存在。只有当这两个按钮都不存在时，我们才会创建新的按钮和选择器。

希望这个修改能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

User

有几个问题：原本的“一键下载”按钮不见了，另外这几个按钮不在同一行上，而且长得不像按钮。

```

""" // Create a MutationObserver to monitor DOM changes
var observer = new MutationObserver(function(mutations)
{ mutations.forEach(function(mutation) { if (mutation.type
=== 'childList') { var generateButton =
document.querySelector('.ModelActionCard_runPic__0I9wi');
if (generateButton && !document.querySelector('.one-
click-download')) { var downloadButton =
document.createElement('div');
downloadButton.innerHTML = '一键下载';
downloadButton.className = 'one-click-download';
downloadButton.onclick = autoDownload;
generateButton.parentNode.insertBefore(downloadButton,
generateButton.nextSibling); observer.disconnect(); } } });
});

"""

```

Bing

对不起，我没有注意到这个问题。你的代码中，"一键下载"按钮被新的按钮替换了，这就是为什么它会消失。你可以修改你的代码，让新的按钮添加到"一键下载"按钮的后面，而不是替换它。

另外，要让按钮看起来像按钮，并且让它们在同一行上，你可以使用CSS。你可以给每个按钮添加一个样式，比如设置背景颜色、边框、边距等，以使它们看起来像按钮。你还可以使用display: inline-block属性，让按钮在同一行上显示。

以下是修改后的代码：

```

// Create a MutationObserver to monitor DOM changes
var observer = new MutationObserver(function(mutations) {
 mutations.forEach(function(mutation) {
 if (mutation.type === 'childList') {
 var generateButton = document.querySelector('.ModelActionCard_runPic__0I9wi');
 if (generateButton && !document.querySelector('.download-images-only') && !document.querySelector('.download-doc-only')) {
 // 创建"仅下载图片"按钮
 var downloadImageButton = document.createElement('button');
 downloadImageButton.innerHTML = '仅下载图片';
 downloadImageButton.className = 'download-images-only';
 downloadImageButton.style = 'margin-left: 10px; display: inline-block;';
 downloadImageButton.onclick = function() {
 var modelName = getModelName();
 var imageCount = document.querySelector('.image-count-selector').value;
 downloadImages(modelName, imageCount);
 };

 // 创建"仅下载文档"按钮
 var downloadDocButton = document.createElement('button');
 downloadDocButton.innerHTML = '仅下载文档';
 downloadDocButton.className = 'download-doc-only';
 downloadDocButton.style = 'margin-left: 10px; display: inline-block;';
 downloadDocButton.onclick = function() {
 saveAsHTML(modelName);
 // saveAsMarkdown(modelName);
 saveAsPlainText(modelName);
 };

 // 创建图片数量选择器
 var imageCountSelector = document.createElement('input');
 imageCountSelector.type = 'number';
 imageCountSelector.min = '1';
 imageCountSelector.value = '10'; // 默认值为10
 imageCountSelector.className = 'image-count-selector';
 imageCountSelector.style = 'margin-left: 10px; display: inline-block;';

 // 将新的按钮和选择器添加到页面上
 generateButton.parentNode.insertBefore(downloadImageButton, generateButton.nextSibling);
 generateButton.parentNode.insertBefore(downloadDocButton, downloadImageButton.nextSibling);
 generateButton.parentNode.insertBefore(imageCountSelector, downloadImageButton.nextSibling);

 observer.disconnect();
 }
 }
 });
});

```

在这个版本的代码中，我使用button元素来创建按钮，而不是div元素，这样按钮会有默认的风格，看起来更像按钮。我还添加了style属性，设置了margin-left（左边距）和display（显示方式）。

希望这个修改能帮到你！如果你还有其他问题，欢迎随时向我提问。😊

User

功能完美实现了，多谢。

Bing

很高兴听到你的功能已经完美实现了！如果你以后还有任何问题或需要帮助，欢迎随时向我提问。祝你一切顺利！😊