* **[Skip to main content](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "content)**
* **[Skip to search](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "main-q)**
* **Technologies**
* **References & Guides**
* **Feedback**

窗体顶端

Search MDN

窗体底端

**[Sign in](https://developer.mozilla.org/zh-CN/signin?next=/zh-CN/docs/Web/API/File/Using_files_from_web_applications)**

1. [Web 开发技术](https://developer.mozilla.org/zh-CN/docs/Web)
2. [Web API 接口参考](https://developer.mozilla.org/zh-CN/docs/Web/API)
3. [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)
4. [在web应用程序中使用文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications)

* [Change language](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "select-language)
* [View in English](https://developer.mozilla.org/en-US/docs/Web/API/File/Using_files_from_web_applications)

[This page was translated from English by the community. Learn more and join the MDN Web Docs community.](https://developer.mozilla.org/en-US/docs/MDN/Contribute/Localize" \l "active_locales)

## Table of contents

* [访问被选择的文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E8%AE%BF%E9%97%AE%E8%A2%AB%E9%80%89%E6%8B%A9%E7%9A%84%E6%96%87%E4%BB%B6)
* [获取被选择文件的信息](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E8%8E%B7%E5%8F%96%E8%A2%AB%E9%80%89%E6%8B%A9%E6%96%87%E4%BB%B6%E7%9A%84%E4%BF%A1%E6%81%AF)
* [通过 click() 方法使用隐藏的 file input 元素](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E9%80%9A%E8%BF%87_click_%E6%96%B9%E6%B3%95%E4%BD%BF%E7%94%A8%E9%9A%90%E8%97%8F%E7%9A%84_file_input_%E5%85%83%E7%B4%A0)
* [使用 label 元素来触发一个隐藏的 file input 元素](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BD%BF%E7%94%A8_label_%E5%85%83%E7%B4%A0%E6%9D%A5%E8%A7%A6%E5%8F%91%E4%B8%80%E4%B8%AA%E9%9A%90%E8%97%8F%E7%9A%84_file_input_%E5%85%83%E7%B4%A0)
* [使用拖放来选择文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BD%BF%E7%94%A8%E6%8B%96%E6%94%BE%E6%9D%A5%E9%80%89%E6%8B%A9%E6%96%87%E4%BB%B6)
* [例子：显示用户选择的图片的缩略图](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E6%98%BE%E7%A4%BA%E7%94%A8%E6%88%B7%E9%80%89%E6%8B%A9%E7%9A%84%E5%9B%BE%E7%89%87%E7%9A%84%E7%BC%A9%E7%95%A5%E5%9B%BE)
* [使用对象 URL](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BD%BF%E7%94%A8%E5%AF%B9%E8%B1%A1_url)
* [示例：使用对象URL来显示图片](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E7%A4%BA%E4%BE%8B%EF%BC%9A%E4%BD%BF%E7%94%A8%E5%AF%B9%E8%B1%A1url%E6%9D%A5%E6%98%BE%E7%A4%BA%E5%9B%BE%E7%89%87)
* [例子：上传一个用户选择的文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E4%B8%8A%E4%BC%A0%E4%B8%80%E4%B8%AA%E7%94%A8%E6%88%B7%E9%80%89%E6%8B%A9%E7%9A%84%E6%96%87%E4%BB%B6)
* [例子：用对象URL显示PDF](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E7%94%A8%E5%AF%B9%E8%B1%A1url%E6%98%BE%E7%A4%BApdf)
* [例子：将对象URL用于其它文件类型](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E5%B0%86%E5%AF%B9%E8%B1%A1url%E7%94%A8%E4%BA%8E%E5%85%B6%E5%AE%83%E6%96%87%E4%BB%B6%E7%B1%BB%E5%9E%8B)
* [规范](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E8%A7%84%E8%8C%83)
* [相关链接](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E7%9B%B8%E5%85%B3%E9%93%BE%E6%8E%A5)

# 在web应用程序中使用文件

通过使用在 HTML5 中加入到 DOM 的 File API，使在web内容中让用户选择本地文件然后读取这些文件的内容成为可能。用户可以通过HTML中的 [<input type="file">](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/input/file" \o "This is a link to an unwritten page) 元素或者是通过拖拽来选择本地文件。

如果你想通过扩展或者其它的chrome源码浏览器（browser chrome code）使用DOM File API，是可行的；然而，需要注意有一些附加特性。详细请见 [Using the DOM File API in chrome code](https://developer.mozilla.org/en/Extensions/Using_the_DOM_File_API_in_chrome_code" \o "This is a link to an unwritten page)。

## [访问被选择的文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E8%AE%BF%E9%97%AE%E8%A2%AB%E9%80%89%E6%8B%A9%E7%9A%84%E6%96%87%E4%BB%B6" \o "Permalink to 访问被选择的文件)

考虑这段 HTML:

<input type="file" id="input">

通过 File API，我们可以访问 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList)，它包含了表示用户所选文件的 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 对象

如果用户只选择了一个文件，那么只需要考虑列表中的第一个文件。

使用传统的 DOM 选择器访问一个被选择的文件：

const selectedFile = document.getElementById('input').files[0];

### [通过 change 事件访问被选择的文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E9%80%9A%E8%BF%87_change_%E4%BA%8B%E4%BB%B6%E8%AE%BF%E9%97%AE%E8%A2%AB%E9%80%89%E6%8B%A9%E7%9A%84%E6%96%87%E4%BB%B6" \o "Permalink to 通过 change 事件访问被选择的文件)

可以（但不是强制的）通过 change 事件访问 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList)：

<input type="file" id="input" onchange="handleFiles(this.files)">

当用户选择一个文件时，handleFiles() 方法会用一个 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList) 对象作为参数被调用，[FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList) 对象包含表示用户选择的文件的 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 对象。

如果你想让用户选择多个文件，只需在 input 元素上使用 multiple 属性：

<input type="file" id="input" multiple onchange="handleFiles(this.files)">

在这个例子中，对于每个用户选择的文件，传递给 handleFiles()方法的文件列表都包含一个对应的 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 对象。

### [动态添加change监听器](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E5%8A%A8%E6%80%81%E6%B7%BB%E5%8A%A0change%E7%9B%91%E5%90%AC%E5%99%A8" \o "Permalink to 动态添加change监听器)

你需要使用 [EventTarget.addEventListener()](https://developer.mozilla.org/zh-CN/docs/Web/API/EventTarget/addEventListener) 去添加 change 事件监听器，像这样：

const inputElement = document.getElementById("input");

inputElement.addEventListener("change", handleFiles, false);function handleFiles() {

const fileList = this.files; /\* now you can work with the file list \*/}

注意在这个例子里，handleFiles() 方法本身是一个事件处理器，不像之前的例子中，它被事件处理器调用然后传递给它一个参数。

## [获取被选择文件的信息](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E8%8E%B7%E5%8F%96%E8%A2%AB%E9%80%89%E6%8B%A9%E6%96%87%E4%BB%B6%E7%9A%84%E4%BF%A1%E6%81%AF" \o "Permalink to 获取被选择文件的信息)

DOM 提供 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList) 对象列出了用户选择的所有文件，每一个文件被指定为一个 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 对象。你可以通过检查文件列表 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList) 的 length 属性来确定用户选择了多少个文件：

const numFiles = files.length;

可以像数组一样简单地访问文件列表来检索各个 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 对象：

for (let i = 0, numFiles = files.length; i < numFiles; i++) {

const file = files[i];

// ...}

这个循环遍历了文件列表里的所有文件。

[File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 对象提供了三个属性，包含了文件的有用信息。

**name**

文件名称，只读字符串。只包含文件名，不包含任何路径信息。

**size**

以字节数为单位的文件大小，只读的64位整数。

**type**

文件的 MIME 类型，只读字符串，当类型不能确定时为 ""。

### [示例：显示文件大小](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E7%A4%BA%E4%BE%8B%EF%BC%9A%E6%98%BE%E7%A4%BA%E6%96%87%E4%BB%B6%E5%A4%A7%E5%B0%8F" \o "Permalink to 示例：显示文件大小)

下面的例子展示了 size 属性的一种可能用法：

<!DOCTYPE html><html><head><meta charset="UTF-8"><title>File(s) size</title><script>function updateSize() {

let nBytes = 0,

oFiles = document.getElementById("uploadInput").files,

nFiles = oFiles.length;

for (let nFileId = 0; nFileId < nFiles; nFileId++) {

nBytes += oFiles[nFileId].size;

}

let sOutput = nBytes + " bytes";

// optional code for multiples approximation

const aMultiples = ["KiB", "MiB", "GiB", "TiB", "PiB", "EiB", "ZiB", "YiB"];

for (nMultiple = 0, nApprox = nBytes / 1024; nApprox > 1; nApprox /= 1024, nMultiple++) {

sOutput = nApprox.toFixed(3) + " " + aMultiples[nMultiple] + " (" + nBytes + " bytes)";

}

// end of optional code

document.getElementById("fileNum").innerHTML = nFiles;

document.getElementById("fileSize").innerHTML = sOutput;}</script></head>

<body onload="updateSize()">

<form name="uploadForm">

<div>

<input id="uploadInput" type="file" name="myFiles" onchange="updateSize();" multiple>

selected files: <span id="fileNum">0</span>;

total size: <span id="fileSize">0</span>

</div>

<div><input type="submit" value="Send file"></div>

</form></body></html>

## [通过 click() 方法使用隐藏的 file input 元素](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E9%80%9A%E8%BF%87_click_%E6%96%B9%E6%B3%95%E4%BD%BF%E7%94%A8%E9%9A%90%E8%97%8F%E7%9A%84_file_input_%E5%85%83%E7%B4%A0" \o "Permalink to 通过 click() 方法使用隐藏的 file input 元素)

你可以隐藏公认难看的 file [<input>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/input" \o "This is a link to an unwritten page) 元素并显示你自己的界面来打开文件选择器，然后显示哪个或哪些文件被用户选中了。你可以通过给 input 元素添加 display:none 的样式，再调用 [<input>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/input" \o "This is a link to an unwritten page) 元素的 click() 方法来实现。

考虑这段HTML：

<input type="file" id="fileElem" multiple accept="image/\*" style="display:none" onchange="handleFiles(this.files)"><button id="fileSelect">Select some files</button>

处理 click 事件的代码类似于这样：

const fileSelect = document.getElementById("fileSelect"),

fileElem = document.getElementById("fileElem");

fileSelect.addEventListener("click", function (e) {

if (fileElem) {

fileElem.click();

}}, false);

你可以给这个用来打开文件选择器的新按钮添加任何你想要的样式。

## [使用 label 元素来触发一个隐藏的 file input 元素](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BD%BF%E7%94%A8_label_%E5%85%83%E7%B4%A0%E6%9D%A5%E8%A7%A6%E5%8F%91%E4%B8%80%E4%B8%AA%E9%9A%90%E8%97%8F%E7%9A%84_file_input_%E5%85%83%E7%B4%A0" \o "Permalink to 使用 label 元素来触发一个隐藏的 file input 元素)

允许在不使用 JavaScript（click() 方法）来打开文件选择器，可以使用 [<label>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/label) 元素。注意在这种情况下，input 元素不能使用 display: none（或 visibility: hidden）隐藏，否则 label 将无法通过键盘访问。使用 [visually-hidden technique](https://a11yproject.com/posts/how-to-hide-content/) 作为替代。

考虑这段HTML：

<input type="file" id="fileElem" multiple accept="image/\*" class="visually-hidden"><label for="fileElem">Select some files</label>

和这段CSS：

.visually-hidden {

position: absolute **!important**;

height: 1px;

width: 1px;

overflow: hidden;

clip: rect(1px, 1px, 1px, 1px);}

/\* Separate rule for compatibility, :focus-within is required on modern Firefox and Chrome \*/input.visually-hidden:focus + label {

outline: thin dotted;}input.visually-hidden:focus-within + label {

outline: thin dotted;}

这里不需要添加任何 JavaScript 代码来调用fileElem.click()，另外，这时你也可以给 label 元素添加你想要的样式。您需要在其 label 上提供隐藏 input 字段的焦点状态的视觉提示，比如上面用的轮廓，或者背景颜色或边框阴影。（截至编写时为止，Firefox 不显示 <input type="file"> 元素的视觉提示。）

## [使用拖放来选择文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BD%BF%E7%94%A8%E6%8B%96%E6%94%BE%E6%9D%A5%E9%80%89%E6%8B%A9%E6%96%87%E4%BB%B6" \o "Permalink to 使用拖放来选择文件)

你还可以让用户将文件拖拽到你的网页应用中。

第一步是创建一个drop区域。虽然你网页内容的哪部分接受拖放取决于你的应用设计，但是使一个元素接收drop事件是很容易的。

let dropbox;

dropbox = document.getElementById("dropbox");

dropbox.addEventListener("dragenter", dragenter, false);

dropbox.addEventListener("dragover", dragover, false);

dropbox.addEventListener("drop", drop, false);

在这个例子中，我们将ID为dropbox的元素变为了我们的drop区域。这是通过给元素添加[dragenter](https://developer.mozilla.org/zh-CN/docs/Web/Reference/Events/dragenter" \o "This is a link to an unwritten page), [dragover](https://developer.mozilla.org/zh-CN/docs/Web/Reference/Events/dragover" \o "This is a link to an unwritten page), 和[drop](https://developer.mozilla.org/zh-CN/docs/Web/Reference/Events/drop" \o "This is a link to an unwritten page) 事件监听器实现的。

我们其实并不需要对dragenter and dragover 事件进行处理，所以这些函数都很简单。他们只需要包括禁止事件传播和阻止默认事件：

function dragenter(e) {

e.stopPropagation();

e.preventDefault();}

function dragover(e) {

e.stopPropagation();

e.preventDefault();}

真正的奥妙在drop()这个函数中：

function drop(e) {

e.stopPropagation();

e.preventDefault();

var dt = e.dataTransfer;

var files = dt.files;

handleFiles(files);}

这里，我们从事件中获取到了dataTransfer 这个域，然后从中得到文件列表，再将它们传递给handleFiles()函数。在这之后，处理文件的方法和用input元素或者用拖拽就是一样的了。

## [例子：显示用户选择的图片的缩略图](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E6%98%BE%E7%A4%BA%E7%94%A8%E6%88%B7%E9%80%89%E6%8B%A9%E7%9A%84%E5%9B%BE%E7%89%87%E7%9A%84%E7%BC%A9%E7%95%A5%E5%9B%BE" \o "Permalink to 例子：显示用户选择的图片的缩略图)

比方说，你正在开发一个炫酷的下一代图片分享网站，并且想使用HTML5来展示用户在实际上传之前的图片的缩略图。你可以像我们之前讨论的那样创建自己的input元素或者drop区域，然后对他们使用一个回调函数，比如下面的handleFiles()。

function handleFiles(files) {

for (var i = 0; i < files.length; i++) {

var file = files[i];

var imageType = /^image\//;

if (!imageType.test(file.type)) {

continue;

}

var img = document.createElement("img");

img.classList.add("obj");

img.file = file;

preview.appendChild(img); // 假设"preview"就是用来显示内容的div

var reader = new FileReader();

reader.onload = (function(aImg) { return function(e) { aImg.src = e.target.result; }; })(img);

reader.readAsDataURL(file);

}}

这里我们循环处理用户选择的文件，看每个文件的type属性是不是image（通过正则表达式来匹配MIME类型字符串模式"image/\*"）。 对每个文件而言，如果它是image，我们就创建一个新的img元素。可以使用css来创建一个漂亮的边框或阴影来显示图片的具体大小，在这儿就不具体做了。

为了在DOM树中更容易地找到他们，每个图片元素都被添加了一个名为obj的CSS类。我们还给每个图片添加了file属性使它具有 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)；这样做可以让我们拿到稍后需要实际上传的图片。我们在预览页中使用 [Node.appendChild()](https://developer.mozilla.org/zh-CN/docs/Web/API/Node/appendChild)来添加新的缩略图。

接下来，我们创建了[FileReader](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReader)来处理异步的图片加载并把他赋给img元素。在创建一个新的 FileReader对象后，我们新建了它的onload 函数，然后调用readAsDataURL()函数开始后台读取文件。当整个图片文件的内容都被全部加载完后，它们被转换成了一个被传递到onload回调函数的data:URL。我们再执行常规操作将img元素的src属性设置为刚刚加载完毕的URL，使得图像可以显示在用户屏幕上的缩略图中。

## [使用对象 URL](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BD%BF%E7%94%A8%E5%AF%B9%E8%B1%A1_url" \o "Permalink to 使用对象 URL)

Gecko 2.0 (Firefox 4 / Thunderbird 3.3 / SeaMonkey 2.1)引入了对DOM [window.URL.createObjectURL() (en-US)](https://developer.mozilla.org/en-US/docs/Web/API/URL/createObjectURL" \o "Currently only available in English (US)) 和 [window.URL.revokeObjectURL() (en-US)](https://developer.mozilla.org/en-US/docs/Web/API/URL/revokeObjectURL" \o "Currently only available in English (US)) 方法的支持。这使得你可以创建用于引用任何数据的简单URL字符串，也可以引用一个包括用户电脑上的本地文件的DOM [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)对象。

当你需要在HTML中通过URL来引用一个[File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)对象时，你可以创建一个对象URL，就像这样：

var objectURL = window.URL.createObjectURL(fileObj);

这个对象URL是一个标识[File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)对象的字符串。每次你调用[window.URL.createObjectURL() (en-US)](https://developer.mozilla.org/en-US/docs/Web/API/URL/createObjectURL" \o "Currently only available in English (US))，就会产生一个唯一的对象URL，即使是你对一个已创建了对象URL的文件再次创建一个对象URL。每个创建了的对象URL必须要释放。当文档关闭时，它们会自动被释放。如果你的网页要动态使用它们，你需要显式调用 [window.URL.revokeObjectURL() (en-US)](https://developer.mozilla.org/en-US/docs/Web/API/URL/revokeObjectURL" \o "Currently only available in English (US))来释放它们：

window.URL.revokeObjectURL(objectURL);

## [示例：使用对象URL来显示图片](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E7%A4%BA%E4%BE%8B%EF%BC%9A%E4%BD%BF%E7%94%A8%E5%AF%B9%E8%B1%A1url%E6%9D%A5%E6%98%BE%E7%A4%BA%E5%9B%BE%E7%89%87" \o "Permalink to 示例：使用对象URL来显示图片)

这个例子使用对象URL来显示图片缩略图。另外，示例也会显示文件名和文件大小等其他文件信息。

显示接口的HTML类似于这样：

<input type="file" id="fileElem" multiple accept="image/\*" style="display:none" onchange="handleFiles(this.files)"><a href="#" id="fileSelect">Select some files</a><div id="fileList">

<p>No files selected!</p></div>

这确定我们的文件 [<input>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/input" \o "This is a link to an unwritten page) 元素显示为一个可以调用文件选择器的链接（我们隐藏了文件输入元素来阻止显示用户不友好的界面）。这个在 [Using hidden file input elements using the click() method](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "using_hidden_file_input_elements_using_the_click()_method)节已经说明了这种调用文件选择器的方法。

handleFiles() 方法如下：

window.URL = window.URL || window.webkitURL;

var fileSelect = document.getElementById("fileSelect"),

fileElem = document.getElementById("fileElem"),

fileList = document.getElementById("fileList");

fileSelect.addEventListener("click", function (e) {

if (fileElem) {

fileElem.click();

}

e.preventDefault(); // prevent navigation to "#"}, false);

function handleFiles(files) {

if (!files.length) {

fileList.innerHTML = "<p>No files selected!</p>";

} else {

fileList.innerHTML = "";

var list = document.createElement("ul");

fileList.appendChild(list);

for (var i = 0; i < files.length; i++) {

var li = document.createElement("li");

list.appendChild(li);

var img = document.createElement("img");

img.src = window.URL.createObjectURL(files[i]);

img.height = 60;

img.onload = function () {

window.URL.revokeObjectURL(this.src);

}

li.appendChild(img);

var info = document.createElement("span");

info.innerHTML = files[i].name + ": " + files[i].size + " bytes";

li.appendChild(info);

}

}}

首先，获取ID为 fileList 的 [<div>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/div) 。这个区块里我们会插入我们的文件列表，包括缩略图。

如果传入 handleFiles() 的 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList) 对象值为 null 时，我们只要简单将这块的内部HTML为显示“No files selected!”。否则，我们就需要像下面这样编写我们的文件列表：

1. 创建一个无序列表 ([<ul>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/ul)) 元素。
2. 通过调用列表的[Node.appendChild()](https://developer.mozilla.org/zh-CN/docs/Web/API/Node/appendChild)方法来将新的列表元素插入到 [<div>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/div)块。
3. 遍历文件集合 [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList)（即files）中的每个 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)：
   1. 创建一个新的列表项（[<li>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/li)）元素并插入到列表中。
   2. 创建一个新的图片（[<img>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/img)）元素。
   3. 设置图片的源为一个新的指代文件的对象URL，使用[window.URL.createObjectURL() (en-US)](https://developer.mozilla.org/en-US/docs/Web/API/URL/createObjectURL" \o "Currently only available in English (US))来创建blob URL。
   4. 设置图片的高度为60像素。
   5. 设置图片的load事件处理器来释放对象URL，当图片加载完成之后对象URL就不再需要了。这个可以通过调用[window.URL.revokeObjectURL() (en-US)](https://developer.mozilla.org/en-US/docs/Web/API/URL/revokeObjectURL" \o "Currently only available in English (US))方法并且传递 img.src中的对象URL字符串来实现。
   6. 将新的列表项添加到列表中。

这是上面代码的一个在线示例：

## [例子：上传一个用户选择的文件](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E4%B8%8A%E4%BC%A0%E4%B8%80%E4%B8%AA%E7%94%A8%E6%88%B7%E9%80%89%E6%8B%A9%E7%9A%84%E6%96%87%E4%BB%B6" \o "Permalink to 例子：上传一个用户选择的文件)

另一件您可能想要做的事是让用户将选定的一个或多个文件（例如前一个示例中选择的图像）上传到服务器。这用异步可以很容易地完成。

### [创建上传任务](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E5%88%9B%E5%BB%BA%E4%B8%8A%E4%BC%A0%E4%BB%BB%E5%8A%A1" \o "Permalink to 创建上传任务)

继续前面示例中构建缩略图的代码，回想一下每个缩略图图像都在CSS类obj中，并且file属性中附加了相应的 [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File) 。这允许我们使用 [Document.querySelectorAll()](https://developer.mozilla.org/zh-CN/docs/Web/API/Document/querySelectorAll) 选择用户决定上传的所有图像，如下所示：

function sendFiles() {

var imgs = document.querySelectorAll(".obj");

for (var i = 0; i < imgs.length; i++) {

new FileUpload(imgs[i], imgs[i].file);

}}

第2行获取了文档中所有CSS类为obj的元素的 [NodeList](https://developer.mozilla.org/zh-CN/docs/Web/API/NodeList)，命名为imgs。在我们的例子中，这些是包含所有图像缩略图的列表。有了这个列表，遍历并为每一项创建一个新的FileUpload实例就很简单了。每个实例都可以处理相应文件的上传。

### [Handling the upload process for a file处理文件的上传过程](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "handling_the_upload_process_for_a_file%E5%A4%84%E7%90%86%E6%96%87%E4%BB%B6%E7%9A%84%E4%B8%8A%E4%BC%A0%E8%BF%87%E7%A8%8B" \o "Permalink to Handling the upload process for a file处理文件的上传过程)

FileUpload函数接受两个输入：一个image元素和一个包含图像数据的文件。

function FileUpload(img, file) {

var reader = new FileReader();

this.ctrl = createThrobber(img);

var xhr = new XMLHttpRequest();

this.xhr = xhr;

var self = this;

this.xhr.upload.addEventListener("progress", function(e) {

if (e.lengthComputable) {

var percentage = Math.round((e.loaded \* 100) / e.total);

self.ctrl.update(percentage);

}

}, false);

xhr.upload.addEventListener("load", function(e){

self.ctrl.update(100);

var canvas = self.ctrl.ctx.canvas;

canvas.parentNode.removeChild(canvas);

}, false);

xhr.open("POST", "http://demos.hacks.mozilla.org/paul/demos/resources/webservices/devnull.php");

xhr.overrideMimeType('text/plain; charset=x-user-defined-binary');

reader.onload = function(evt) {

xhr.send(evt.target.result);

};

reader.readAsBinaryString(file);}

上面的FileUpload() 函数创建了一个“加载中”指示器，用于显示进度信息，然后创建了一个 [XMLHttpRequest](https://developer.mozilla.org/zh-CN/docs/Web/API/XMLHttpRequest) 来处理上传数据。

实际传输数据前，采取了几道准备步骤：

1. XMLHttpRequest的progress监听器被设为将加载指示器更新为新的百分比信息，这样随着上传进行，指示器会显示最新的信息。
2. XMLHttpRequest的load事件监听器被设为将加载指示器的进度信息更新为100%，以保证进度显示确实达到了100%（以防在上传过程中出现粒度误差）。然后它移除了已经不再需要的加载指示器。这样上传一完成指示器就会消失。
3. 上传图像文件的请求，是由调用XMLHttpRequest的open()函数发送POST请求完成的。
4. 上传的MIME类型是通过调用XMLHttpRequest的overrideMimeType()函数来设置的。这个例子中，我们使用通用MIME类型。是否需要设置MIME类型取决于你的具体使用情况。
5. FileReader对象用于将文件转换为二进制字符串。
6. 最后，当内容被加载时，会调用XMLHttpRequest的send()函数来上传文件内容。

**提示：**上面例子中使用的非标准的sendAsBinary方法在Gecko 31 (Firefox 31 / Thunderbird 31 / SeaMonkey 2.28) 中已废弃。请使用标准的send(Blob data)方法代替。

### [异步处理文件上传](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E5%BC%82%E6%AD%A5%E5%A4%84%E7%90%86%E6%96%87%E4%BB%B6%E4%B8%8A%E4%BC%A0" \o "Permalink to 异步处理文件上传)

这个例子演示了如何异步上传文件，在服务器端使用了php、在客户端使用了JavaScript。

<?phpif (isset($\_FILES['myFile'])) {

// Example:

move\_uploaded\_file($\_FILES['myFile']['tmp\_name'], "uploads/" . $\_FILES['myFile']['name']);

exit;}?><!DOCTYPE html><html><head>

<title>dnd binary upload</title>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<script type="application/javascript">

function sendFile(file) {

const uri = "/index.php";

const xhr = new XMLHttpRequest();

const fd = new FormData();

xhr.open("POST", uri, true);

xhr.onreadystatechange = function() {

if (xhr.readyState == 4 && xhr.status == 200) {

alert(xhr.responseText); // handle response.

}

};

fd.append('myFile', file);

// Initiate a multipart/form-data upload

xhr.send(fd);

}

window.onload = function() {

const dropzone = document.getElementById("dropzone");

dropzone.ondragover = dropzone.ondragenter = function(event) {

event.stopPropagation();

event.preventDefault();

}

dropzone.ondrop = function(event) {

event.stopPropagation();

event.preventDefault();

const filesArray = event.dataTransfer.files;

for (let i=0; i<filesArray.length; i++) {

sendFile(filesArray[i]);

}

}

}

</script></head><body>

<div>

<div id="dropzone" style="margin:30px; width:500px; height:300px; border:1px dotted grey;">Drag & drop your file here...</div>

</div></body></html>

## [例子：用对象URL显示PDF](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E7%94%A8%E5%AF%B9%E8%B1%A1url%E6%98%BE%E7%A4%BApdf" \o "Permalink to 例子：用对象URL显示PDF)

对象URL可以用于image之外的其它东西！它可以用于显示嵌入的PDF文件或任何其它浏览器能显示的资源。

在Firefox中，要让PDF嵌入式地显示在iframe中（而不是作为下载的文件弹出），必须将pdfjs.disabled设为false .

<iframe id="viewer">

这是src属性的改动：

const obj\_url = window.URL.createObjectURL(blob);const iframe = document.getElementById('viewer');

iframe.setAttribute('src', obj\_url);

window.URL.revokeObjectURL(obj\_url);

## [例子：将对象URL用于其它文件类型](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E4%BE%8B%E5%AD%90%EF%BC%9A%E5%B0%86%E5%AF%B9%E8%B1%A1url%E7%94%A8%E4%BA%8E%E5%85%B6%E5%AE%83%E6%96%87%E4%BB%B6%E7%B1%BB%E5%9E%8B" \o "Permalink to 例子：将对象URL用于其它文件类型)

你可以用同样方式操作其它格式的文件。这是预览上传的视频的方法：

const video = document.getElementById('video');const obj\_url = window.URL.createObjectURL(blob);

video.src = obj\_url;

video.play()

window.URL.revokeObjectURL(obj\_url);

## [规范](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E8%A7%84%E8%8C%83" \o "Permalink to 规范)

| **规范** | **状态** | **注解** |
| --- | --- | --- |
| [HTML Living Standard File upload state](https://html.spec.whatwg.org/multipage/number-state.html" \l "concept-input-type-file-selected) | Living Standard |  |
| [File API](https://w3c.github.io/FileAPI/" \o "File API) | Working Draft | Initial definition |

## [相关链接](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications" \l "%E7%9B%B8%E5%85%B3%E9%93%BE%E6%8E%A5" \o "Permalink to 相关链接)

* [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)
* [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList)
* [FileReader](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReader)
* [URL](https://developer.mozilla.org/zh-CN/docs/Web/API/URL)
* [XMLHttpRequest](https://developer.mozilla.org/zh-CN/docs/Web/API/XMLHttpRequest)
* [Using XMLHttpRequest](https://developer.mozilla.org/en/DOM/XMLHttpRequest/Using_XMLHttpRequest" \o "This is a link to an unwritten page)

#### Found a problem with this page?

* [Source on](https://github.com/mdn/translated-content/blob/main/files/zh-cn/web/api/file/using_files_from_web_applications/index.html" \o "Folder: zh-cn/web/api/file/using_files_from_web_applications (Opens in a new tab)" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)**[GitHub](https://github.com/mdn/translated-content/blob/main/files/zh-cn/web/api/file/using_files_from_web_applications/index.html" \o "Folder: zh-cn/web/api/file/using_files_from_web_applications (Opens in a new tab)" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)**
* [Report a problem with this content on](https://github.com/mdn/content/issues/new?body=MDN+URL:+https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications%0A%0A%23%23%23%23+What+information+was+incorrect,+unhelpful,+or+incomplete?%0A%0A%0A%23%23%23%23+Specific+section+or+headline?%0A%0A%0A%23%23%23%23+What+did+you+expect+to+see?%0A%0A%0A%23%23%23%23+Did+you+test+this?+If+so,+how?%0A%0A%0A<!--+Do+not+make+changes+below+this+line+-->%0A<details>%0A<summary>MDN+Content+page+report+details</summary>%0A%0A*+Folder:+`zh-cn/web/api/file/using_files_from_web_applications`%0A*+MDN+URL:+https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications%0A*+GitHub+URL:+https://github.com/mdn/translated-content/blob/main/files/zh-cn/web/api/file/using_files_from_web_applications/index.html%0A*+Last+commit:+https://github.com/mdn/translated-content/commit/ee1c8483d9e05042a81cc812c8d91fe8a1356dff%0A*+Document+last+modified:+2021-05-26T16:26:18.000Z%0A%0A</details>&title=Issue+with+"%E5%9C%A8web%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F%E4%B8%AD%E4%BD%BF%E7%94%A8%E6%96%87%E4%BB%B6":+(short+summary+here+please)&labels=Content:WebAPI,needs-triage" \o "This will take you to https://github.com/mdn/content to file a new issue" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)**[GitHub](https://github.com/mdn/content/issues/new?body=MDN+URL:+https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications%0A%0A%23%23%23%23+What+information+was+incorrect,+unhelpful,+or+incomplete?%0A%0A%0A%23%23%23%23+Specific+section+or+headline?%0A%0A%0A%23%23%23%23+What+did+you+expect+to+see?%0A%0A%0A%23%23%23%23+Did+you+test+this?+If+so,+how?%0A%0A%0A<!--+Do+not+make+changes+below+this+line+-->%0A<details>%0A<summary>MDN+Content+page+report+details</summary>%0A%0A*+Folder:+`zh-cn/web/api/file/using_files_from_web_applications`%0A*+MDN+URL:+https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications%0A*+GitHub+URL:+https://github.com/mdn/translated-content/blob/main/files/zh-cn/web/api/file/using_files_from_web_applications/index.html%0A*+Last+commit:+https://github.com/mdn/translated-content/commit/ee1c8483d9e05042a81cc812c8d91fe8a1356dff%0A*+Document+last+modified:+2021-05-26T16:26:18.000Z%0A%0A</details>&title=Issue+with+"%E5%9C%A8web%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F%E4%B8%AD%E4%BD%BF%E7%94%A8%E6%96%87%E4%BB%B6":+(short+summary+here+please)&labels=Content:WebAPI,needs-triage" \o "This will take you to https://github.com/mdn/content to file a new issue" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)**
* Want to fix the problem yourself? See [our Contribution guide](https://github.com/mdn/content/blob/main/README.md" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank).

**Last modified:** 2021年5月27日, [by MDN contributors](https://developer.mozilla.org/zh-CN/docs/Web/API/File/Using_files_from_web_applications/contributors.txt)

窗体顶端

Change your languageSelect your preferred language Change language

窗体底端

#### Related Topics

1. [File](https://developer.mozilla.org/zh-CN/docs/Web/API/File)
2. Constructor
   1. [File()](https://developer.mozilla.org/zh-CN/docs/Web/API/File/File)
3. 属性
   1. [fileName](https://developer.mozilla.org/zh-CN/docs/Web/API/File/fileName)
   2. [fileSize](https://developer.mozilla.org/zh-CN/docs/Web/API/File/fileSize)
   3. [lastModified](https://developer.mozilla.org/zh-CN/docs/Web/API/File/lastModified)
   4. [lastModifiedDate](https://developer.mozilla.org/zh-CN/docs/Web/API/File/lastModifiedDate)
   5. [mozFullPath](https://developer.mozilla.org/zh-CN/docs/Web/API/File/mozFullPath)
   6. [name](https://developer.mozilla.org/zh-CN/docs/Web/API/File/name)
   7. [type](https://developer.mozilla.org/zh-CN/docs/Web/API/File/type)
   8. [webkitRelativePath](https://developer.mozilla.org/zh-CN/docs/Web/API/File/webkitRelativePath)
4. 方法
   1. [getAsBinary()](https://developer.mozilla.org/zh-CN/docs/Web/API/File/getAsBinary)
   2. [getAsDataURL()](https://developer.mozilla.org/zh-CN/docs/Web/API/File/getAsDataURL)
   3. [getAsText()](https://developer.mozilla.org/zh-CN/docs/Web/API/File/getAsText)
5. 继承
   1. [Blob](https://developer.mozilla.org/zh-CN/docs/Web/API/Blob)
6. Related pages for File API
   1. [Blob](https://developer.mozilla.org/zh-CN/docs/Web/API/Blob)
   2. [FileList](https://developer.mozilla.org/zh-CN/docs/Web/API/FileList)
   3. [FileReader](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReader)
   4. [FileReaderSync](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReaderSync)

* [Web Technologies](https://developer.mozilla.org/zh-CN/docs/Web)
* [Learn Web Development](https://developer.mozilla.org/zh-CN/docs/Learn)
* [About MDN](https://developer.mozilla.org/zh-CN/docs/MDN/About)
* [Feedback](https://developer.mozilla.org/zh-CN/docs/MDN/Feedback)
* [About](https://www.mozilla.org/about/" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [MDN Web Docs Store](https://shop.spreadshirt.com/mdn-store/" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [Contact Us](https://www.mozilla.org/contact/" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [Firefox](https://www.mozilla.org/firefox/?utm_source=developer.mozilla.org&utm_campaign=footer&utm_medium=referral" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)

#### MDN

* [MDN on Twitter](https://twitter.com/mozdevnet" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [MDN on Github](https://github.com/mdn/" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)

#### Mozilla

* [Mozilla on Twitter](https://twitter.com/mozilla" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [Mozilla on Instagram](https://www.instagram.com/mozillagram/" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)

© 2005-2021 Mozilla and individual contributors. Content is available under [these licenses](https://developer.mozilla.org/docs/MDN/About" \l "Copyrights_and_licenses).

* [Terms](https://www.mozilla.org/about/legal/terms/mozilla" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [Privacy](https://www.mozilla.org/privacy/websites/" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)
* [Cookies](https://www.mozilla.org/privacy/websites/" \l "cookies" \t "https://developer.mozilla.org/zh-CN/docs/Web/API/File/_blank)