Использование файлов из веб-приложений

<color #ed1c24>Продолжение https://developer.mozilla.org/en-US/docs/Web/API/File_API/Using_files_from_web_applications</color>

Используя File API, веб-контент может попросить пользователя выбрать локальные файлы, а затем прочитать их содержимое. Этот выбор можно сделать либо с помощью <input type="file"> элемента HTML, либо путем перетаскивания.

Доступ к выбранным файлам

Рассмотрим этот HTML:

HTML

```
<input type="file" id="input" multiple />
```

API файлов позволяет получить доступ к FileList содержащимся File объектам, представляющим файлы, выбранные пользователем.

Атрибут multipleэлемента inputпозволяет пользователю выбирать несколько файлов.

Доступ к первому выбранному файлу с помощью классического селектора DOM:

JS

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

Доступ к выбранным файлам в случае изменения

Также возможно (но не обязательно) получить доступ FileList через change событие. Вам нужно использовать EventTarget.addEventListener() для добавления changeпрослушивателя событий, например:

JS

```
const inputElement = document.getElementById("input");
inputElement.addEventListener("change", handleFiles, false);
function handleFiles() {
  const fileList = this.files; /* now you can work with the file list
*/
}
```

Получение информации о выбранных файлах

Объект FileList, предоставленный DOM, перечисляет все файлы, выбранные пользователем, каждый из которых указан как объект File. Вы можете определить, сколько файлов выбрал пользователь, проверив значение атрибута списка файлов length:

JS

```
const numFiles = fileList.length;
```

Отдельные File объекты можно получить, обратившись к списку как к массиву.

Объект предоставляет три атрибута File, которые содержат полезную информацию о файле.

- name

Имя файла в виде строки, доступной только для чтения. Это просто имя файла, не содержащее никакой информации о пути.

- size

Размер файла в байтах как 64-битное целое число, доступное только для чтения.

- type

MIME-тип файла в виде строки, доступной только для чтения, или «» если тип не может быть определен.

Пример: отображение размера файла(ов)

В следующем примере показано возможное использование свойства size:

HTML

```
<!doctype html>
<html lang="en-US">
  <head>
    <meta charset="UTF-8" />
    <title>File(s) size</title>
  </head>

  <body>
    <form name="uploadForm">
      <div>
        <input id="uploadInput" type="file" multiple />
        <label for="fileNum">Selected files:</label>
        <output id="fileNum">0</output>;
```

```
        <label for="fileSize">Total size:</label>
        <output id="fileSize">0</output>
      </div>
      <div><input type="submit" value="Send file" /></div>
    </form>

    <script>
      const uploadInput = document.getElementById("uploadInput");
      uploadInput.addEventListener(
        "change",
        () => {
          // Calculate total size
          let numberOfBytes = 0;
          for (const file of uploadInput.files) {
            numberOfBytes += file.size;
          }

          // Approximate to the closest prefixed unit
          const units = [
            "B",
            "KiB",
            "MiB",
            "GiB",
            "TiB",
            "PiB",
            "EiB",
            "ZiB",
            "YiB",
          ];
          const exponent = Math.min(
            Math.floor(Math.log(numberOfBytes) / Math.log(1024)),
            units.length - 1,
          );
          const approx = numberOfBytes / 1024 ** exponent;
          const output =
            exponent === 0
              ? `${numberOfBytes} bytes`
              : `${approx.toFixed(3)} ${
                  units[exponent]
                } (${numberOfBytes} bytes)`;

          document.getElementById("fileNum").textContent =
            uploadInput.files.length;
          document.getElementById("fileSize").textContent = output;
        },
        false,
      );
    </script>
  </body>
</html>
```

Использование скрытых элементов ввода файла с помощью метода click()

Вы можете скрыть заведомо уродливый файловый <input> элемент и предоставить свой собственный интерфейс для открытия средства выбора файлов и отображения файла или файлов, выбранных пользователем. Вы можете сделать это, задав стиль входному элементу `display:none` и вызвав click() метод этого <input> элемента.

Код, обрабатывающий `click` событие, может выглядеть так:

JS

```
const fileSelect = document.getElementById("fileSelect");
const fileElem = document.getElementById("fileElem");

fileSelect.addEventListener(
  "click",
  (e) => {
    if (fileElem) {
      fileElem.click();
    }
  },
  false,
);
```

Вы можете стилизовать <button> как пожелаете.

Продолжение
https://developer.mozilla.org/en-US/docs/Web/API/File_API/Using_files_from_web_applications