



- [`DragEvent`](#)
- [`DataTransfer`](#)
- [`DataTransfer`](#)
 - [`DataTransfer.dropEffect`](#)
 - [`DataTransfer.effectAllowed`](#)
 - [`DataTransfer.files`](#)
 - [`DataTransfer.types`](#)
 - [`DataTransfer.items`](#)
- [`DataTransfer`](#)
 - [`DataTransfer.setData\(\)`](#)
 - [`DataTransfer.getData\(\)`](#)
 - [`DataTransfer.clearData\(\)`](#)
 - [`DataTransfer.setDragImage\(\)`](#)



 drag



☐ ☐ ☐ ☐ true ☐

draggable

```
<div draggable="true">
  □□□□□□
</div>
```

div

draggablefalse

draggable true

- drag
- dragstart target
- dragend ESC target
- dragenter dragend target
- dragover dragenter dragover target
- dragleave target
- drop ESC

```
div.addEventListener('dragstart', function (e) {
  this.style.backgroundColor = 'red';
}, false);

div.addEventListener('dragend', function (e) {
  this.style.backgroundColor = 'green';
}, false);
```

div

```

/* HTML 構造
<div class="dropzone">
  <div id="draggable" draggable="true">
    〇〇〇〇〇〇
  </div>
</div>
<div class="dropzone"></div>
<div class="dropzone"></div>
<div class="dropzone"></div>
*/

// 変数宣言
var dragged;

document.addEventListener('dragstart', function (event) {
  // 初期化
  dragged = event.target;
  // 透明度を半透明にする
  event.target.style.opacity = 0.5;
}, false);

document.addEventListener('dragend', function (event) {
  // 透明度を元に戻す
  event.target.style.opacity = '';
}, false);

document.addEventListener('dragover', function (event) {
  // 既定の動作を防止
  event.preventDefault();
}, false);

document.addEventListener('dragenter', function (event) {
  // 初期化
  // 透明度を半透明にする
  if (event.target.className === 'dropzone') {
    event.target.style.background = 'purple';
  }
}, false);

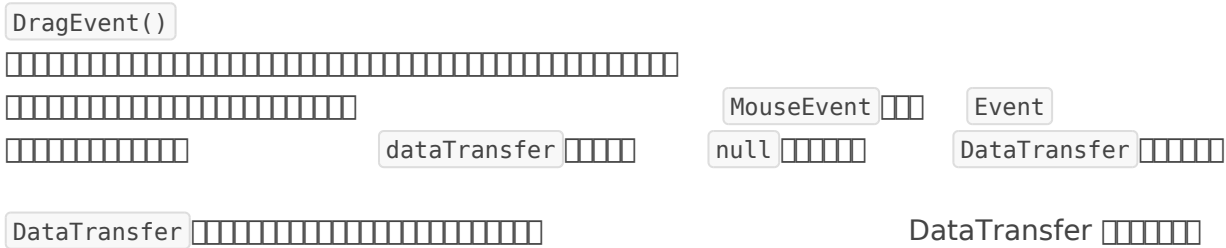
document.addEventListener('dragleave', function( event ) {

```

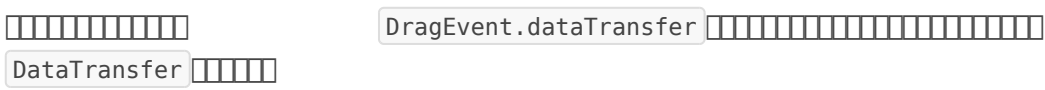
- ```
<div ondragover="return false">
<div ondragover="event.preventDefault()">
```

# DragEvent

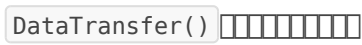
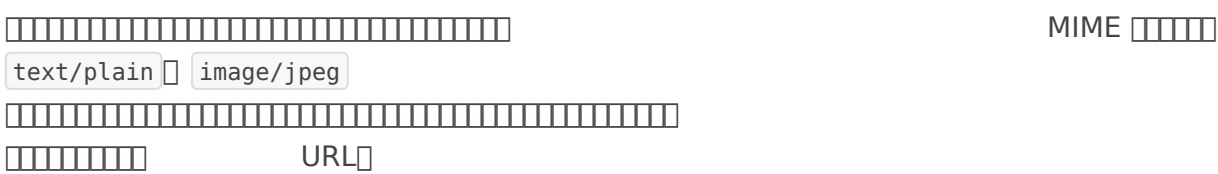
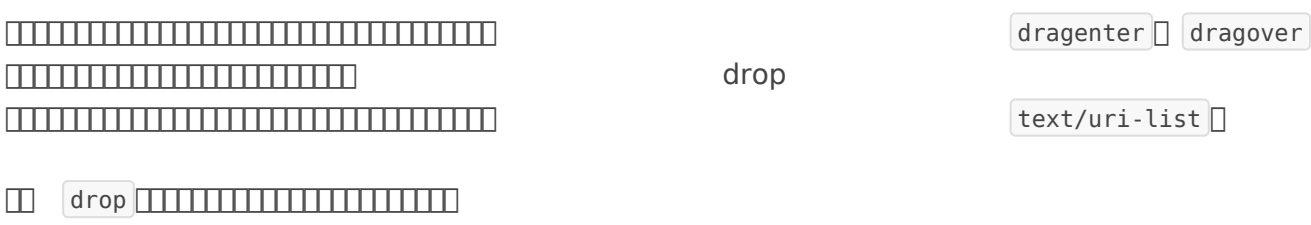
```
new DragEvent(type, options)
```



# DataTransfer

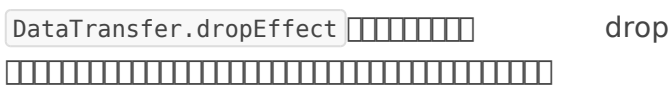


```
var dataTrans = new DataTransfer();
```



# DataTransfer

## DataTransfer.dropEffect



- copy
- move
- link
- none

```
target.addEventListener('dragover', function (e) {
 e.preventDefault();
 e.stopPropagation();
 e.dataTransfer.dropEffect = 'copy';
});
```

**drop**

`dropEffect` [ ]    `draggable` [ ]    `dragover` [ ]

dragstart □ drag □ dragleave

[illegible]

# DataTransfer.effectAllowed

Transfer.effectAllowed

- copy[ ]
- move[ ]
- link[ ]
- copyLink[ ]    copy[ ]    link[ ]
- copyMove[ ]    copy[ ]    move[ ]
- linkMove[ ]    link[ ]    move[ ]
- all[ ]
- none[ ]
- uninitialized[ ]    all

Diagram illustrating a sequence of 20 small rectangles. The first 4 rectangles are grouped together and labeled "dropEffect".

--	--	--	--	--

`dragstart`

```
source.addEventListener('dragstart', function (e) {
 e.dataTransfer.effectAllowed = 'move';
});
```

```
target.addEventListener('dragover', function (e) {
 e.dataTransfer.dropEffect = 'move';
});
```

☐ dropEffect ☐ effectAllowed ☐ none ☐ drop

# DataTransfer.files



The diagram illustrates the structure of `DataTransfer.files` and `FileList`. It shows a variable `DataTransfer.files` pointing to a `FileList` object. The `FileList` object is represented as an array of `File` objects. The first `File` object is shown with its `name` property set to `image001.jpg`. The `FileList` object is also shown with its `length` property set to `1`.

```
// HTML
// <div id="output" style="min-height: 200px;border: 1px solid black;">
// □□□□□□□
// </div>

var div = document.getElementById('output');

div.addEventListener("dragenter", function(event) {
 div.textContent = '';
 event.stopPropagation();
 event.preventDefault();
}, false);

div.addEventListener("dragover", function(event) {
 event.stopPropagation();
 event.preventDefault();
}, false);

div.addEventListener("drop", function(event) {
 event.stopPropagation();
 event.preventDefault();
 var files = event.dataTransfer.files;
 for (var i = 0; i < files.length; i++) {
 div.textContent += files[i].name + ' ' + files[i].size + '□□\n';
 }
}, false);
```

```

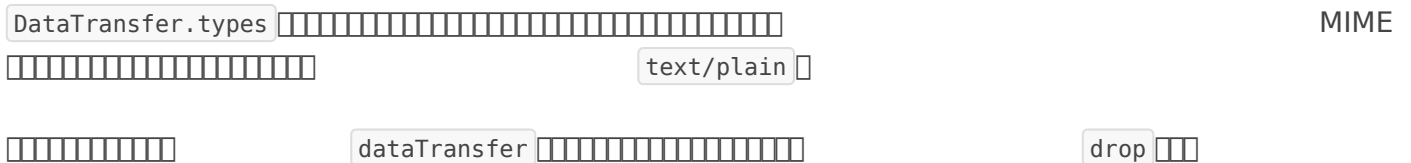
graph LR
 A["dataTransfer.files
(List of 10 File objects)"] --> C["FileUpload"]
 B["FileReader (File object)"] --> C

```

```
div.addEventListener('drop', function(e) {
 e.preventDefault();
 e.stopPropagation();

 var fileList = e.dataTransfer.files;
 if (fileList.length > 0) {
 var file = fileList[0];
 var reader = new FileReader();
 reader.onloadend = function(e) {
 if (e.target.readyState === FileReader.DONE) {
 var content = reader.result;
 div.innerHTML = 'File: ' + file.name + '\n\n' + content;
 }
 }
 reader.readAsBinaryString(file);
 }
});
```

## DataTransfer.types



```
function contains(list, value){
 for (var i = 0; i < list.length; ++i) {
 if(list[i] === value) return true;
 }
 return false;
}

function doDragOver(event) {
 var isLink = contains(event.dataTransfer.types, 'text/uri-list');
 if (isLink) event.preventDefault();
}
```



# DataTransfer.items

DataTransfer.items   
  


DataTransferItemList  
DataTransferItem

DataTransferItemList 

- length 
- add(data, type)  text/html  text/plain 
- add(file)  add 
- remove(index) 
- clear() 

DataTransferItem 

- kind  string  file 
- type  MIME 
- getAsFile()  null 
- getAsString(callback)  
  




```
div.addEventListener('drop', function (e) {
 e.preventDefault();
 if (e.dataTransfer.items != null) {
 for (var i = 0; i < e.dataTransfer.items.length; i++) {
 console.log(e.dataTransfer.items[i].kind + ': ' + e.dataTransfer.items[i].type);
 }
 }
});
```

## DataTransfer

## DataTransfer.setData()

DataTransfer.setData() 

```
event.dataTransfer.setData('text/plain', 'Text to drag');
```



```
text/plain
```

dataTransfer

dataTransfer

```
<div draggable="true">
 aaa
</div>
```

aaa

&lt;/div&gt;

<div>

[illegible]aaa 

Diagram illustrating a memory layout. A box labeled "setData" is followed by a row of 10 empty boxes, representing a memory buffer or array.

setData

[illegible]

```
<div
 draggable="true"
 ondragstart="event.dataTransfer.setData('text/plain', 'bbb')"
>
 aaa
</div>
```

```
draggable="true"
```

```
ondragstart="event.dataTransfer.setData('text/plain', 'bbb')"
```

 $\triangleright$ 

aaa

&lt;/div&gt;

bbb    aaa 

```
text/plain
```

[illegible]

```
var dt = event.dataTransfer;

// 設定
dt.setData('text/uri-list', 'http://www.example.com');
dt.setData('text/plain', 'http://www.example.com');

// HTML 設定
dt.setData('text/html', 'Hello there, stranger');
dt.setData('text/plain', 'Hello there, stranger');

// 設定 URL
```

// ☐ ☐ ☐ ☐

```
dt.setData('text/uri-list', 'http://www.example.com');
```

```
dt.setData('text/plain', 'http://www.example.com');
```

```
// HTML
```

```
dt.setData('text/html', 'Hello there, stranger');
```

```
dt.setData('text/plain', 'Hello there, stranger');
```

// 任意 URL

```
dt.setData('text/uri-list', imageurl);
dt.setData('text/plain', imageurl);
```

```
var dt = event.dataTransfer;

dt.setData('application/x-bookmark', bookmarkString);
dt.setData('text/uri-list', 'http://www.example.com');
dt.setData('text/plain', 'http://www.example.com');
```

drop

[illegible]

drop

[illegible]

# DataTransfer.getData()

```
DataTransfer.getData()
```

[illegible]

drop

setData

drop

drop

```
function onDrop(event) {
 var data = event.dataTransfer.getData('text/plain');
 event.target.textContent = data;
 event.preventDefault();
}
```

[illegible][illegible]

`getData`

```
function doDrop(event) {
 var lines = event.dataTransfer.getData('text/uri-list').split('\n');
 for (let line of lines) {
 let link = document.createElement('a');
 link.href = line;
 link.textContent = line;
 event.target.appendChild(link);
 }
}
```

```
event.preventDefault();
}
```

□□□□□      `getData` □□□□□□□□□□□□□□□□

□□□□□      `URL` □□□□□□□□□□

```
var link = event.dataTransfer.getData('URL');
```

□□□□□□□□□□□□□□□□

```
function doDrop(event) {
 var types = event.dataTransfer.types;
 var supportedTypes = ['text/uri-list', 'text/plain'];
 types = supportedTypes.filter(function (value) { types.includes(value) });
 if (types.length) {
 var data = event.dataTransfer.getData(types[0]);
 }
 event.preventDefault();
}
```

## DataTransfer.clearData()

`DataTransfer.clearData()`

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□  
□□□□□□□□□□□□□□□□□□□□

```
event.dataTransfer.clearData('text/uri-list');
```

□□□□□□□□      `text/uri-list` □□□□□

□□□□□□□□□□□□□□□□□□  
□□□□□□□□□□

`DataTransfer.types` □□□□□□□□

`Files`

□□□□□□□□      `dragstart` □□□□□□□□□□□□□□□□□□□□□□□□□□□□

## DataTransfer.setDragImage()

□□□□□      `dragstart`

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□  
□□□□□□□□□□□□□□□□

`DataTransfer.setDragImage()`

`<img>`

`<canvas>`

`null`

```
/* HTML 测试
```

```
<div id="drag-with-image" class="dragdemo" draggable="true">
```

```
 drag me
```

```
</div>
```

```
*/
```

```
var div = document.getElementById('drag-with-image');
```

```
div.addEventListener('dragstart', function (e) {
```

```
 var img = document.createElement('img');
```

```
 img.src = 'http://path/to/img';
```

```
 e.dataTransfer.setDragImage(img, 0, 0);
```

```
}, false);
```

---

#2

2018-05-08 12:18:12 UTC

2021-02-12 17:37:57 UTC