

HTML 5 Drag & Drop

HTML



Bok, Jong Soon
javaexpert@nate.com

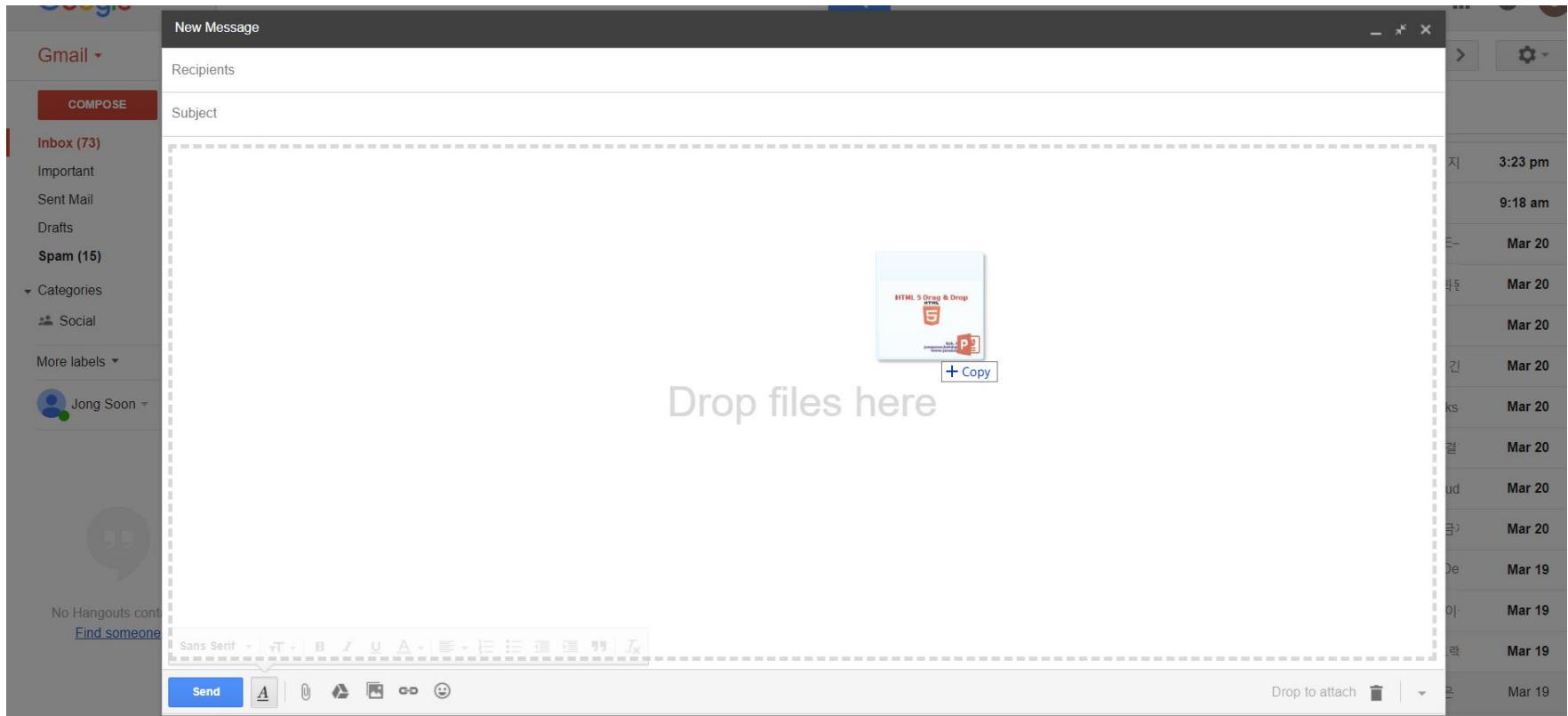
<https://github.com/swacademy/HTML5>

HTML5 Drag & Drop



- Is a part of the HTML5 standard.
- Microsoft IE added drag and drop back in 1999 in IE5.
- Safari had implemented IE's API.
- All Web Browsers support this API.
- Is a very common feature.
- It is when you "grab" an object and drag it to a different location.
- <http://w3c.github.io/html/editing.html#dnd>

HTML5 Drag & Drop (Cont.)



Gmail's file drag & drop

HTML5 Drag & Drop (Cont.)

What's your Name?

Short answer text

Which do you like season?

Multiple choice

- Spring
- Fall
- Summer
- Other...

The image shows a Google Docs interface. On the left, there is a sidebar with icons for adding text, tables, images, videos, and equations. The main content area contains a "Short answer text" field with the placeholder "What's your Name?". Below it is a "Multiple choice" question with the text "Which do you like season?". The options are "Spring", "Fall", "Summer" (which is selected), and "Other...". Each option has a delete button (X) and a media icon.

Google docs Drag & Drop

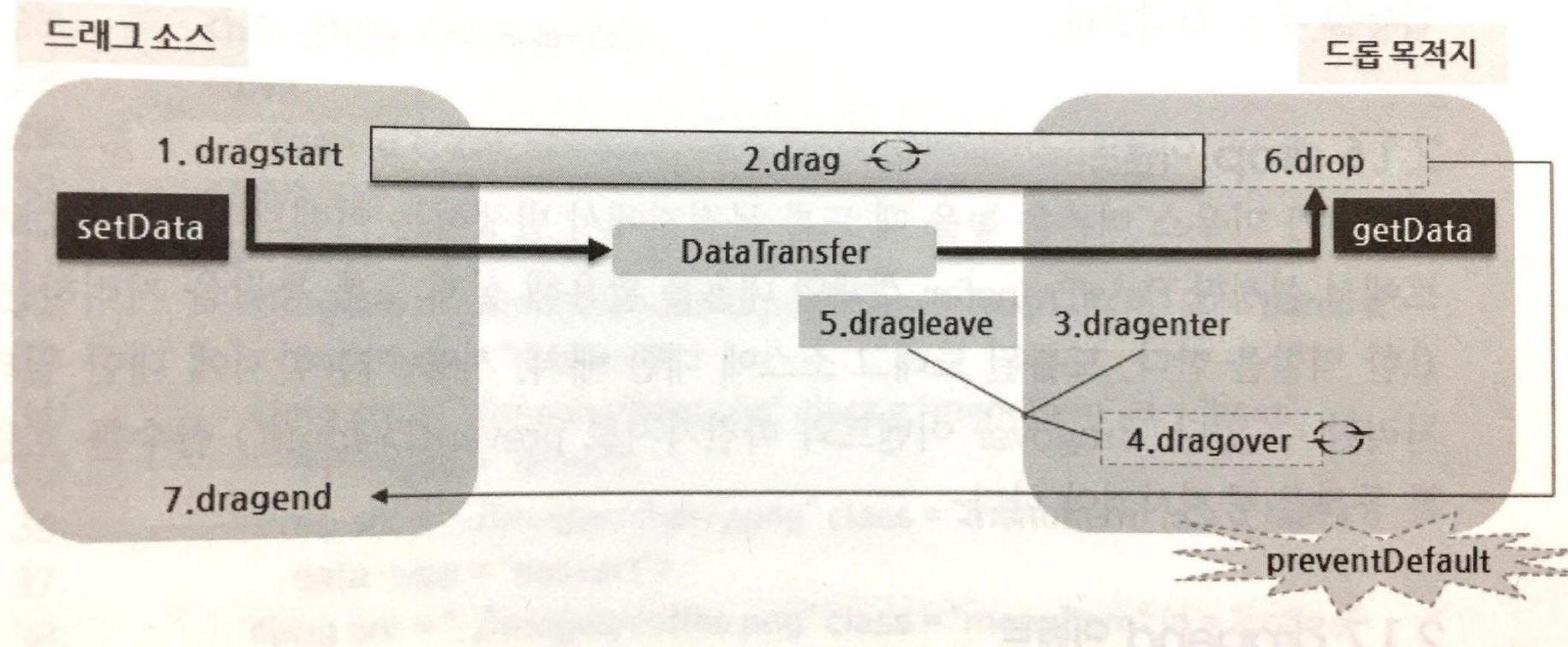
Object's draggable Test

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Drag & Drop Demo</title>
7 </head>
8 <body>
9   
10  <script>
11    var flag = 'draggable' in document.createElement('img');
12    document.write('<br />' + flag + '<br />');
13  </script>
14 </body>
15 </html>
```

Events

| Event Name | Target | Action |
|------------------|--|---|
| dragstart | Source node | Initiate the drag & drop operation |
| drag | Source node | Continue the drag & drop operation |
| dragenter | Immediate user selection or the body element | Reject immediate user selection as potential target element |
| dragexit | Previous target element | None |
| dragleave | Previous target element | None |
| dragover | Current target element | Reset the current drag operation to “none” |
| drop | Current target element | Varies |
| dragend | Source node | Varies |

Events (Cont.)



HTML5 Drag & Drop Process

1. Make an element **draggable**.

- To make an element **draggable**, set the **draggable** attribute to **true**.

```
<img draggable="true">
```

HTML5 Drag & Drop Process (Cont.)

2. What to drag

- Specify what should happen when the element is dragged.
- The **ondragstart** attribute calls a function.
- Specifies what data to be dragged.
- The **dataTransfer.setData()** method sets the data type and the value of the dragged data.

```
function drag(evt) {  
    evt.dataTransfer.setData("Text",  
                           ev.target.id);  
}
```

HTML5 Drag & Drop Process (Cont.)

3. What to drop

- The **ondragover** event specifies where the dragged data can be dropped.
- By default, data/elements cannot be dropped in other elements.
- To allow a drop, we must prevent the default handling of the element.

`event.preventDefault();`

HTML5 Drag & Drop Process (Cont.)

4. Do the Drop

- When the dragged data is dropped, a drop event occurs.
- The **ondrop** attribute calls a function.

```
function drop(evt) {  
    evt.preventDefault();  
    var data=ev.dataTransfer.getData("Text");  
    evt.target.appendChild(  
        document.getElementById(data))  
}
```

Lab1 : Drag & Drop

■ Web Browsers

- IE11, Firefox, Google Chrome, Opera, Safari

■ Text Editors

- Notepad++, Editplus, Visual Studio Code

■ Files

- dragdrop.html
- images/check.png
- js/dragdrop.js

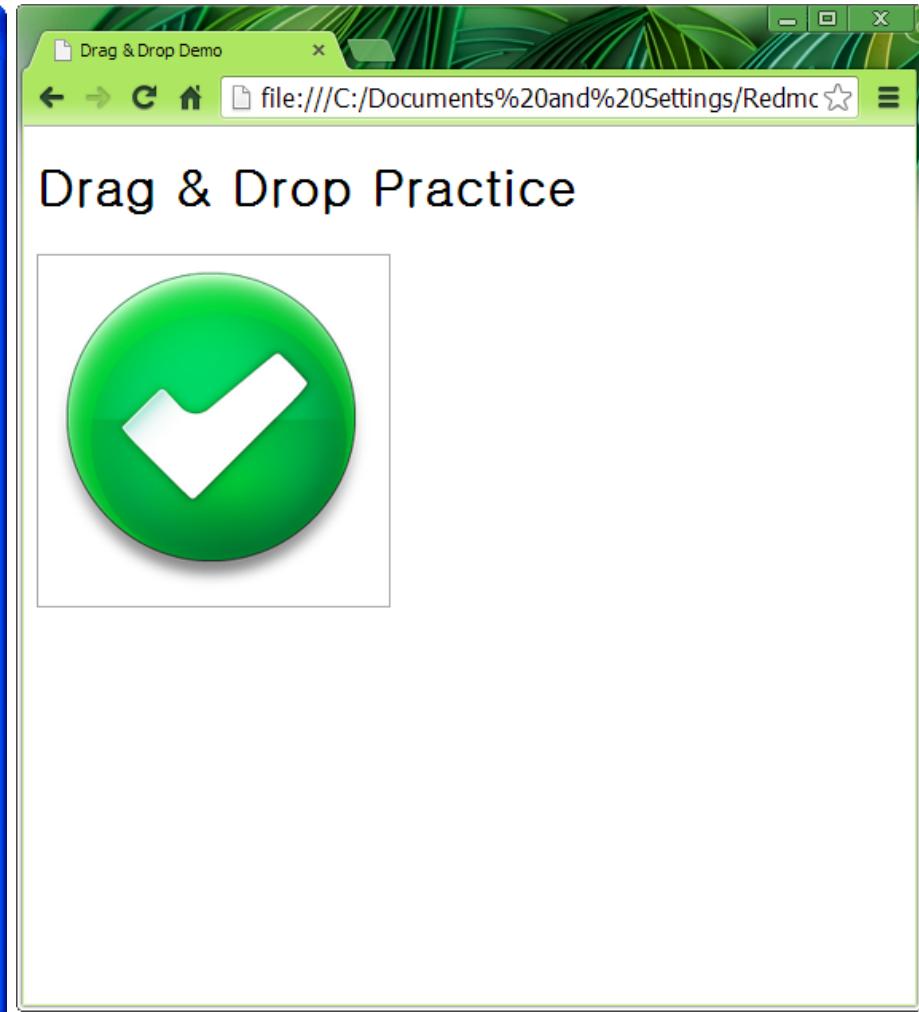
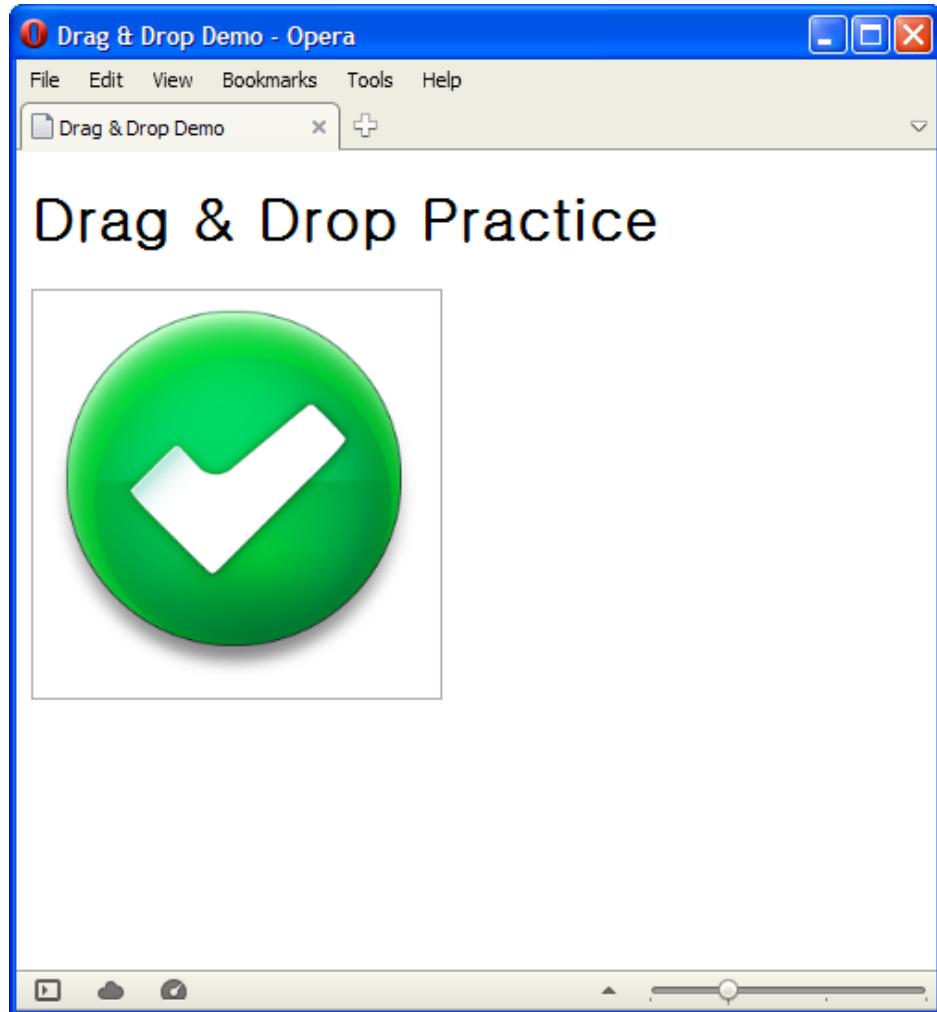
Lab1 : dragdrop.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Drag & Drop Demo</title>
7   <script src='js/dragdrop.js'></script>
8   <style type='text/css'>
9     #bar {
10       width:200px; height:200px; padding:10px; border:1px solid #aaaaaa;
11     }
12   </style>
13 </head>
14 <body>
15   <h1>Drag & Drop Practice</h1>
16   <div id="bar"></div>
17   <br />
18   
20 </body>
21 </html>
```

Lab1 : dragdrop.js

```
1 window.addEventListener('load', setup, false);
2 function setup(){
3     var foo = document.getElementById('foo');
4     foo.addEventListener('dragstart', function(evt){
5         evt.dataTransfer.setData("Text", this.id);
6     }, false);
7     var bar = document.getElementById('bar');
8     bar.addEventListener('drop', function(evt){
9         evt.preventDefault();
10        var id = evt.dataTransfer.getData("Text");
11        var elem = document.getElementById(id);
12        elem.parentNode.removeChild(elem);
13        this.appendChild(elem);
14    }, false);
15    bar.addEventListener('dragover', function(evt){
16        evt.preventDefault();
17    }, false);
18 }
```

Lab1 : Result



The DataTransfer Interface's attributes

■ dropEffect

- Returns the kind of operation that is currently selected.
- `none | copy | link | move`

■ effectAllowed

- Returns the kinds of operations that are to be allowed.
- Can be set to change the allowed operations.
- `none | copy | copyLink | copyMove | link | linkMove | move | all | uninitialized`

The DataTransfer Interface's attributes (Cont.)

■ items

- Returns a **DataTransferItemList** object, with the drag data.

■ types

- Returns an array listing the formats that were set in the dragstart event.
- In addition, if any files are being dragged, then one of the types will be the string "Files".

■ files

- Returns a FileList of the files being dragged, if any.

The DataTransfer Interface's Methods

■ **setDragImage(element, x, y)**

- Uses the given element to update the drag feedback, replacing any previously specified feedback.

■ **getData(format)**

- Returns the specified data.
- If there is no such data, returns the empty string.

■ **setData(format, data)**

- Adds the specified data.

The DataTransfer Interface's Methods (Cont.)

■ **setData (format, data)**

- Adds the specified data.

■ **clearData ([format])**

- Removes the data of the specified formats.
- Removes all data if the argument is omitted.

The **draggable** Attribute

- Returns **true** if the element is draggable; otherwise, returns **false**.
- Can be set, to override the default and set the draggable content attribute.
- The **true** state means the element is draggable; the **false** state means that it is not.
- The **auto** state uses the default behavior of the user agent.

Lab2 : Drag & Drop

■ Web Browsers

- IE10, Firefox, Google Chrome, Opera, Safari

■ Text Editors

- Notepad++ or Editplus

■ Files

- dragdrop1.html
- images/check.png
- js/dragdrop1.js
- css/dragdrop1.css

Lab2 : dragdrop1.html

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title> Drag & Drop Demo </title>
5     <meta charset="utf-8">
6     <link rel="stylesheet" type="text/css" href="css/dragdrop1.css">
7     <script src="js/dragdrop1.js"></script>
8   </head>
9   <body>
10    <div class="bar">
11      
13    </div>
14    <div class="bar" />
15  </body>
16</html>
```

Lab2 : js/dragdrop1.js

```
1 window.addEventListener('load', setup, false);
2 function setup(){
3     var foo = document.getElementById('foo');
4     foo.addEventListener('dragstart', function(evt){
5         evt.dataTransfer.setData("Text", this.id);
6     }, false);
7     var bar = document.getElementsByClassName('bar');
8     for(var i = 0 ; i < bar.length ; i++){
9         bar[i].addEventListener('drop', function(evt){
10            evt.preventDefault();
11            var id = evt.dataTransfer.getData("Text");
12            var elem = document.getElementById(id);
13            elem.parentNode.removeChild(elem);
14            this.appendChild(elem);
15        }, false);
16        bar[i].addEventListener('dragover', function(evt){
17            evt.preventDefault();
18        }, false);
19    }
20 }
```

Lab2 : dragdrop1.css

```
1 #div1, #div2 {  
2     float:left; width:200px;  
3     height:200px; margin:10px;  
4     padding:10px; border:1px solid #aaaaaaaa;  
5 }
```

Lab2 : Result

