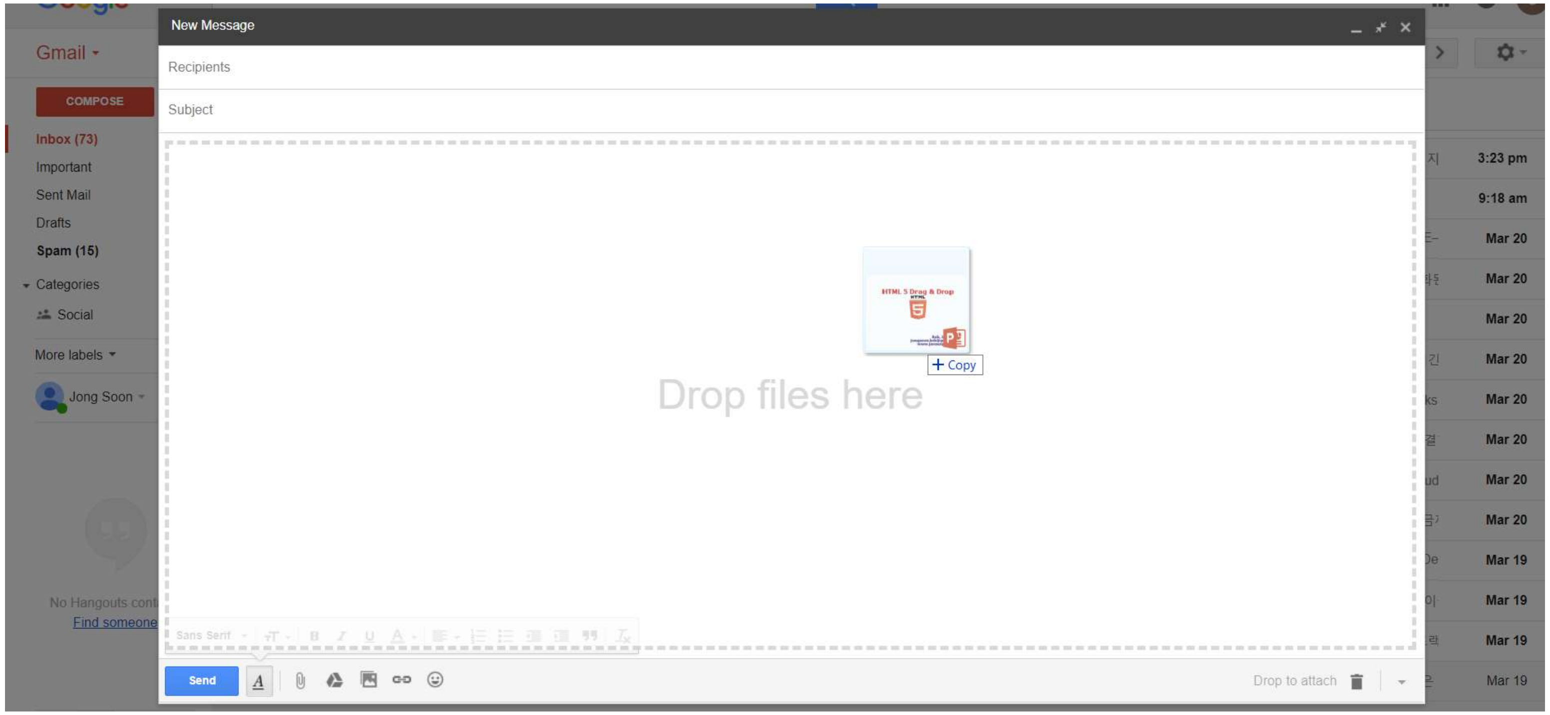
HTML 5 Drag & Drop

Bok, Jong Soon
javaexpert@nate.com
https://github.com/swacademy/HTML5

HTML5 Drag & Drop

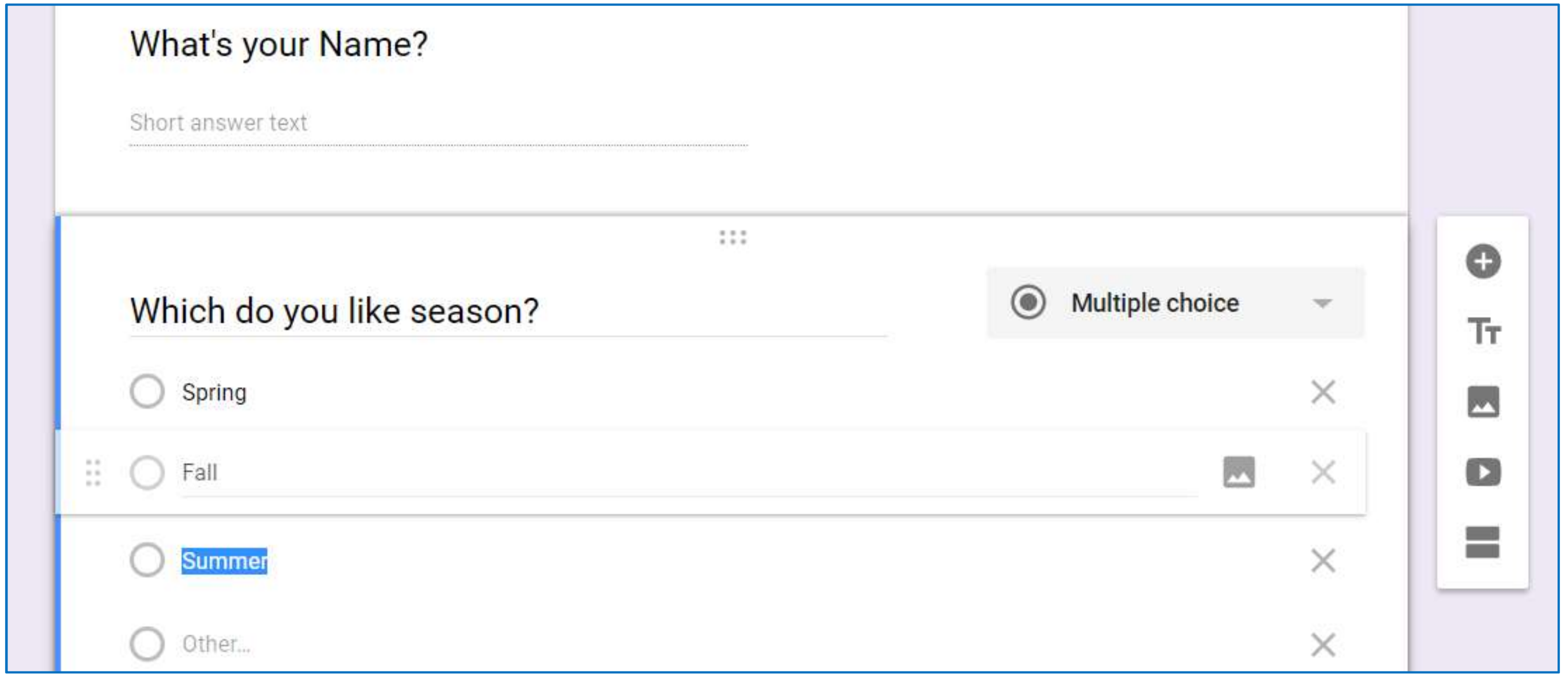
- Is a part of the HTML5 standard.
- Microsoft IE added drag and drop back in 1999 in IE5.
- Safari had implemented IE’s API.
- All Web Browsers support this API.
- Is a very common feature.
- It is when you "grab" an object and drag it to a different location.
- http://w3c.github.io/html/editing.html#dnd

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...

Google docs Drag & Drop

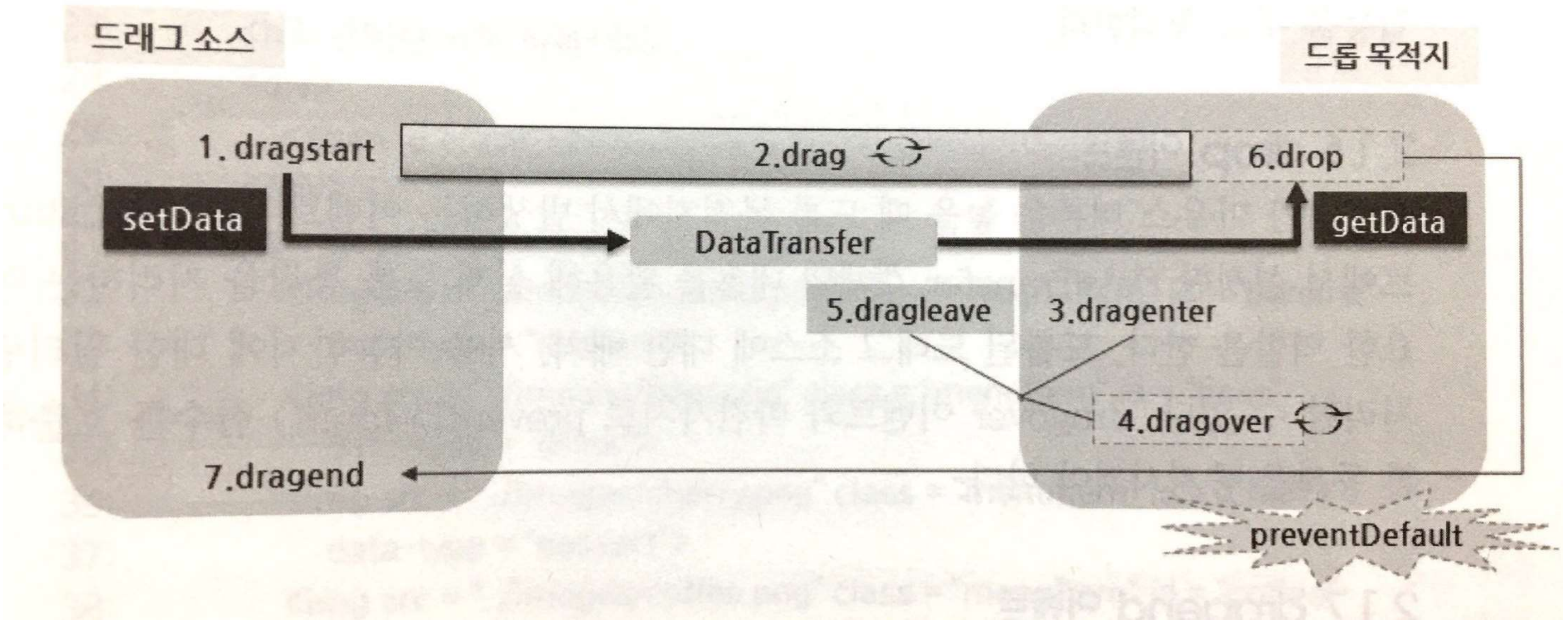
Object’s draggable Test

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Drag &amp; Drop Demo</title>
</head>
<body>
    <img src="images/check.png" alt="check image">
    <script>
        var flag = 'draggable' in document.createElement('img');
        document.write('<br />' + flag + '<br />');
    </script>
</body>
</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.)



조용준, “HTML5 API 프로그래밍”, (서울:가메출판사, 2017), p.203.

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.)

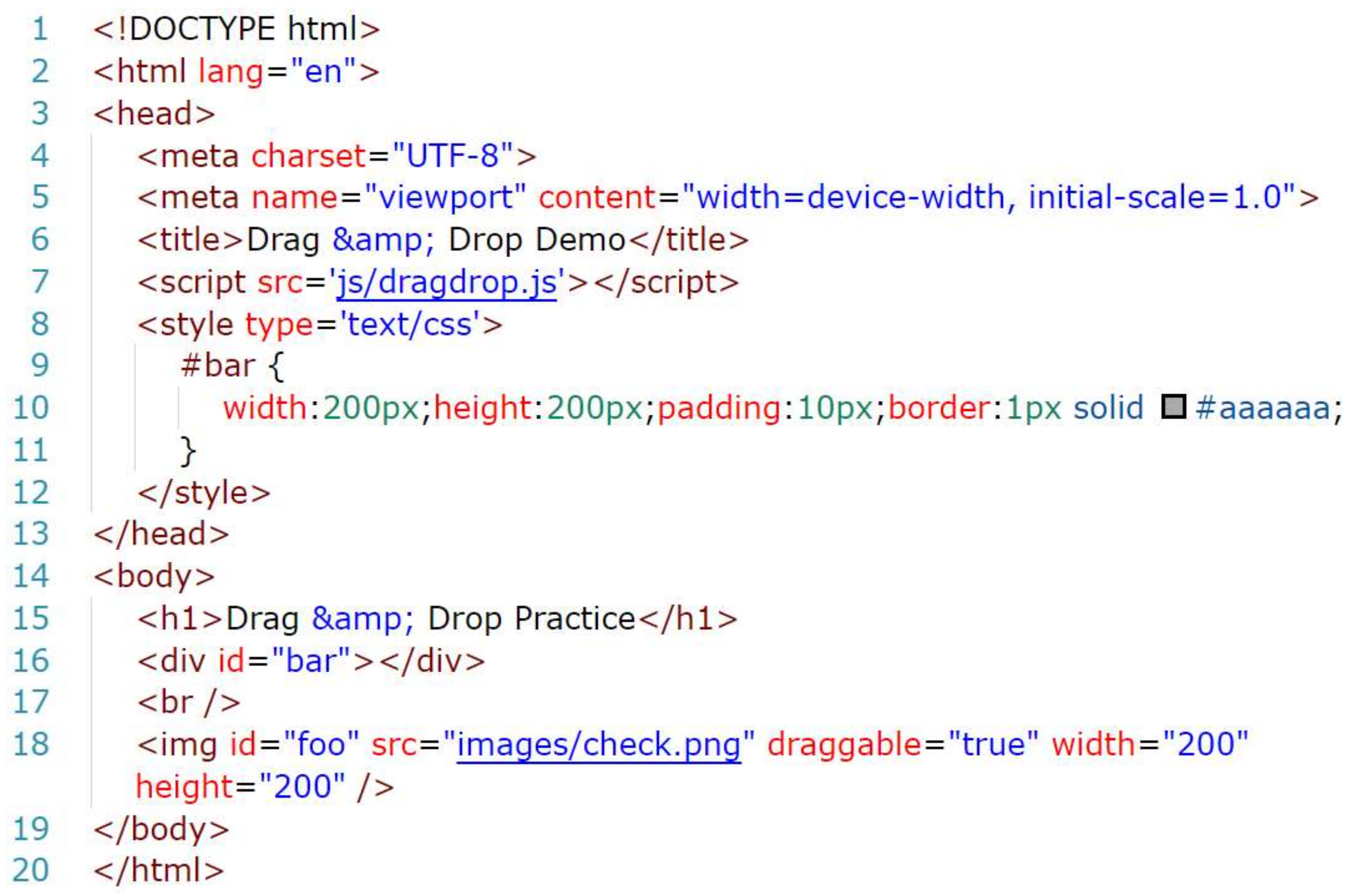
4. Do the Drop
   - When the dragged data is dropped, a drop event occurs.
   - The `ondrop` attribute calls a function.

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

Lab1 : Drag & Drop

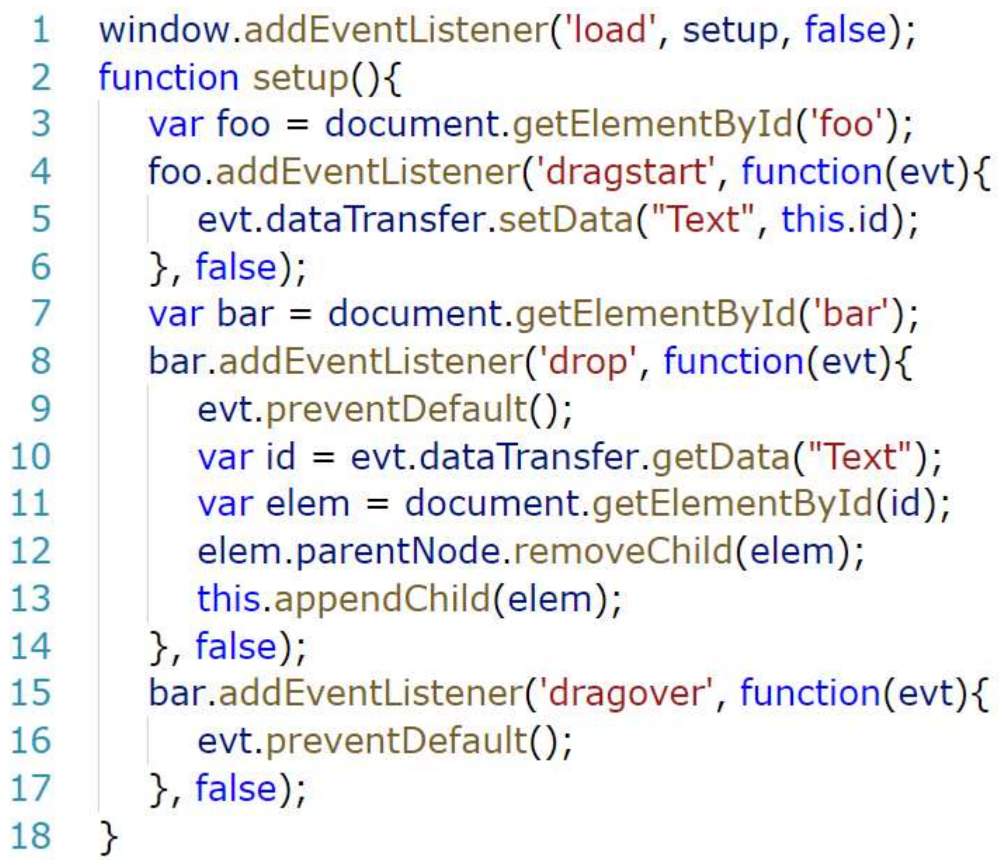
- Web Browsers
  - Edge, Firefox, Google Chrome, Opera, Safari
- Text Editors
  - Visual Studio Code, Notepad++, Editplus, Visual Studio Code
- Files
  - dragdrop.html
  - images/check.png
  - js/dragdrop.js

Lab1 : dragdrop.html



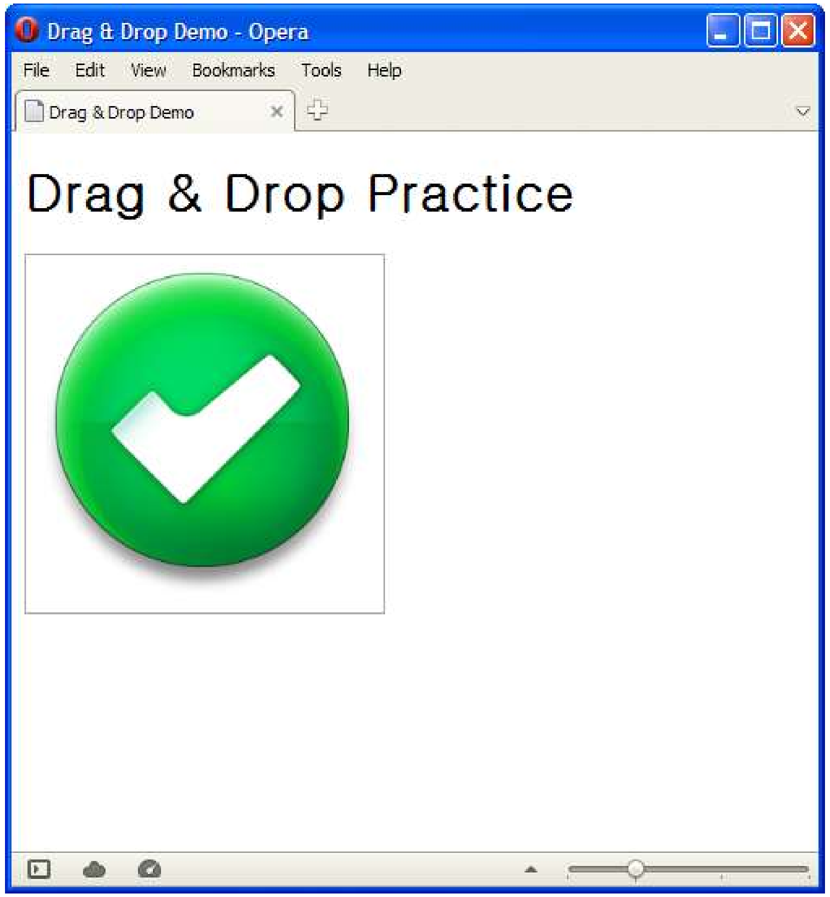
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Drag &amp; Drop Demo</title>
    <script src='js/dragdrop.js'></script>
    <style type='text/css'>
        #bar {
            width:200px;height:200px;padding:10px;border:1px solid #aaaaaa;
        }
    </style>
</head>
<body>
    <h1>Drag &amp; Drop Practice</h1>
    <div id="bar"></div>
    <br />
    <img id="foo" src="images/check.png" draggable="true" width="200" height="200" />
</body>
</html>
```

Lab1 : dragdrop.js

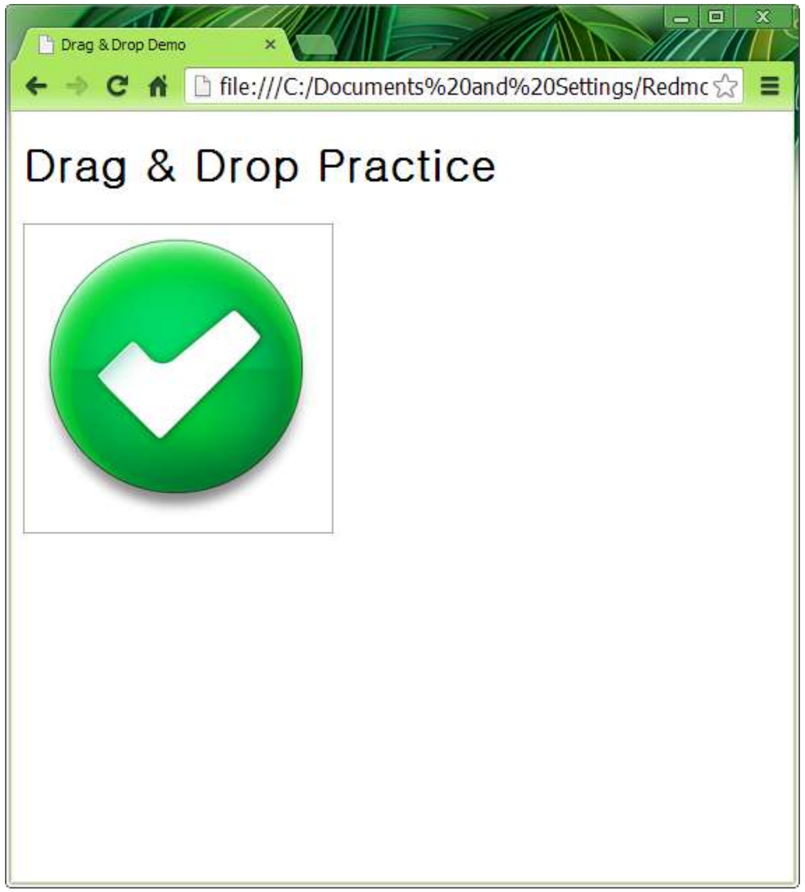


```javascript
window.addEventListener('load', setup, false);
function setup(){
    var foo = document.getElementById('foo');
    foo.addEventListener('dragstart', function(evt){
        evt.dataTransfer.setData("Text", this.id);
    }, false);
    var bar = document.getElementById('bar');
    bar.addEventListener('drop', function(evt){
        evt.preventDefault();
        var id = evt.dataTransfer.getData("Text");
        var elem = document.getElementById(id);
        elem.parentNode.removeChild(elem);
        this.appendChild(elem);
    }, false);
    bar.addEventListener('dragover', function(evt){
        evt.preventDefault();
    }, false);
}
```

Lab1 : Result



Drag & Drop Practice



Drag & Drop Practice

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 e vent.
  - 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.

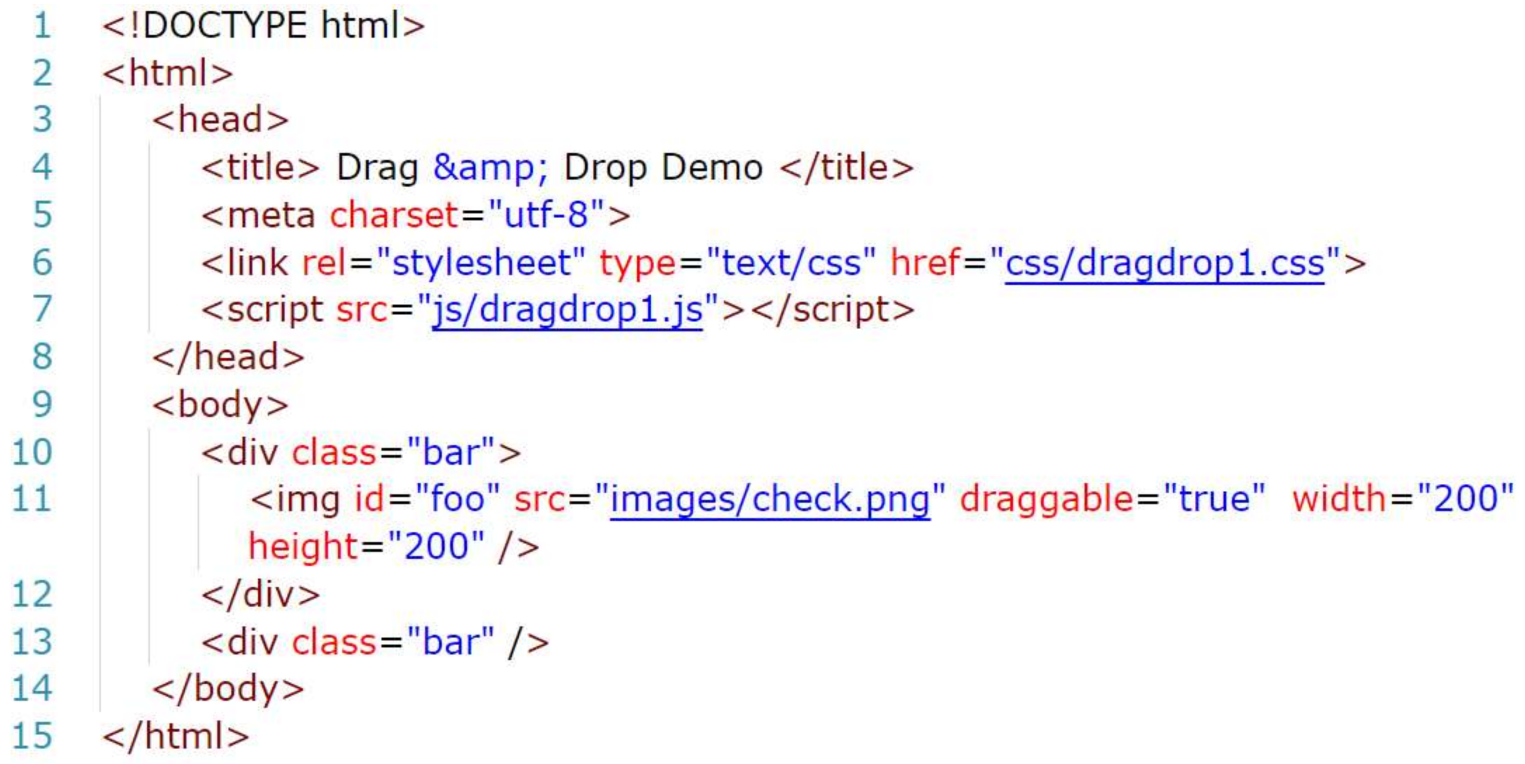
The `draggable` Attribute

- 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

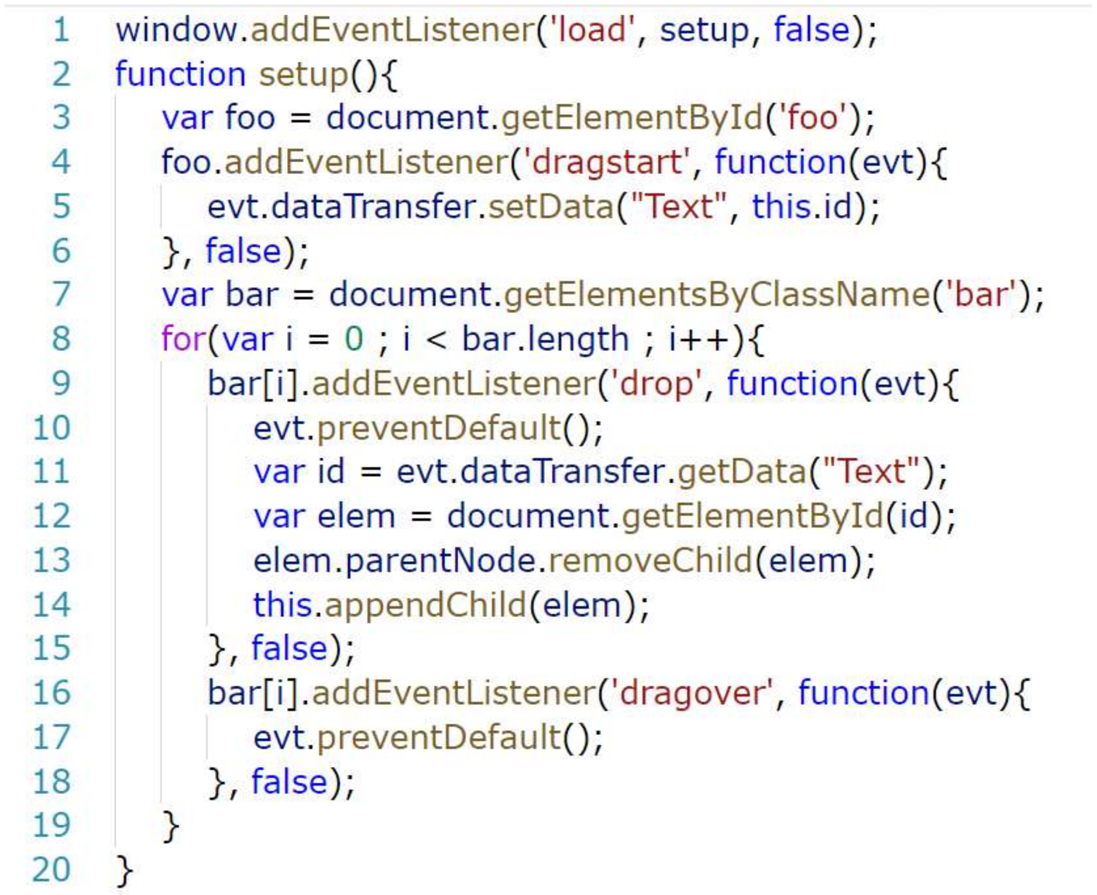
- Web Browsers
  - Edge, Firefox, Google Chrome, Opera, Safari
- Text Editors
  - Visual Studio Code, Notepad++ or Editplus
- Files
  - dragdrop1.html
  - images/check.png
  - js/dragdrop1.js
  - css/dragdrop1.css

Lab2 : dragdrop1.html



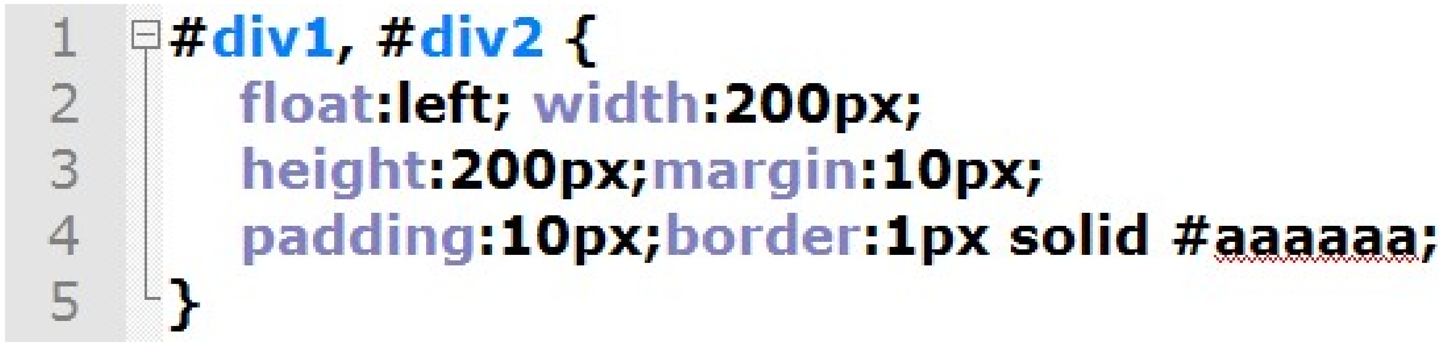
```html
<!DOCTYPE html>
<html>
    <head>
        <title> Drag &amp; Drop Demo </title>
        <meta charset="utf-8">
        <link rel="stylesheet" type="text/css" href="css/dragdrop1.css">
        <script src="js/dragdrop1.js"></script>
    </head>
    <body>
        <div class="bar">
            <img id="foo" src="images/check.png" draggable="true"  width="200" height="200" />
        </div>
        <div class="bar" />
    </body>
</html>
```

Lab2 : js/dragdrop1.js



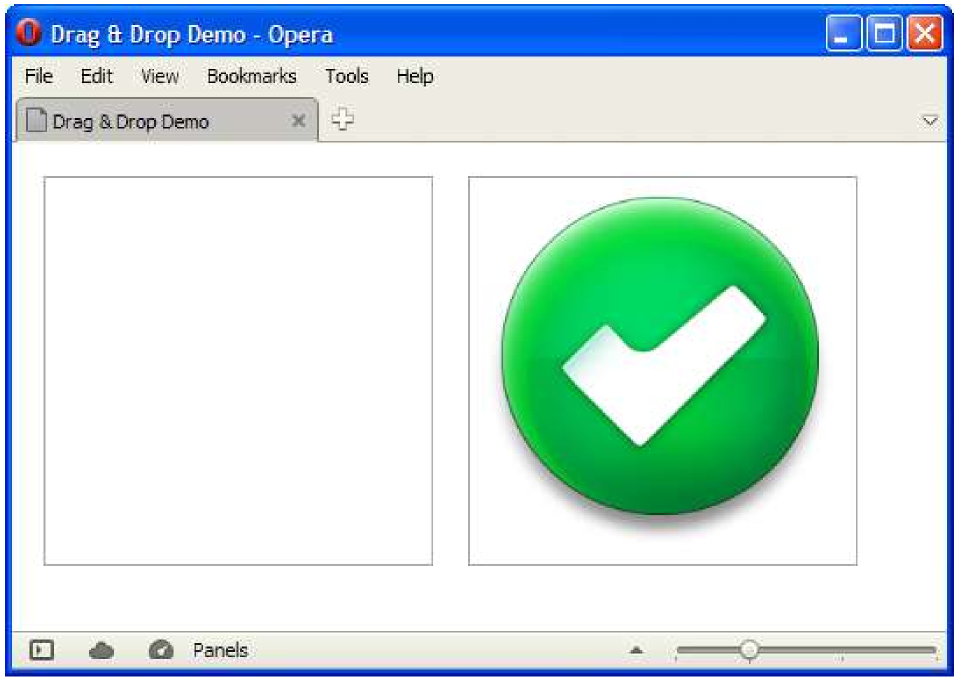
```javascript
window.addEventListener('load', setup, false);
function setup(){
    var foo = document.getElementById('foo');
    foo.addEventListener('dragstart', function(evt){
        evt.dataTransfer.setData("Text", this.id);
    }, false);
    var bar = document.getElementsByClassName('bar');
    for(var i = 0 ; i < bar.length ; i++){
        bar[i].addEventListener('drop', function(evt){
            evt.preventDefault();
            var id = evt.dataTransfer.getData("Text");
            var elem = document.getElementById(id);
            elem.parentNode.removeChild(elem);
            this.appendChild(elem);
        }, false);
        bar[i].addEventListener('dragover', function(evt){
            evt.preventDefault();
        }, false);
    }
}
```

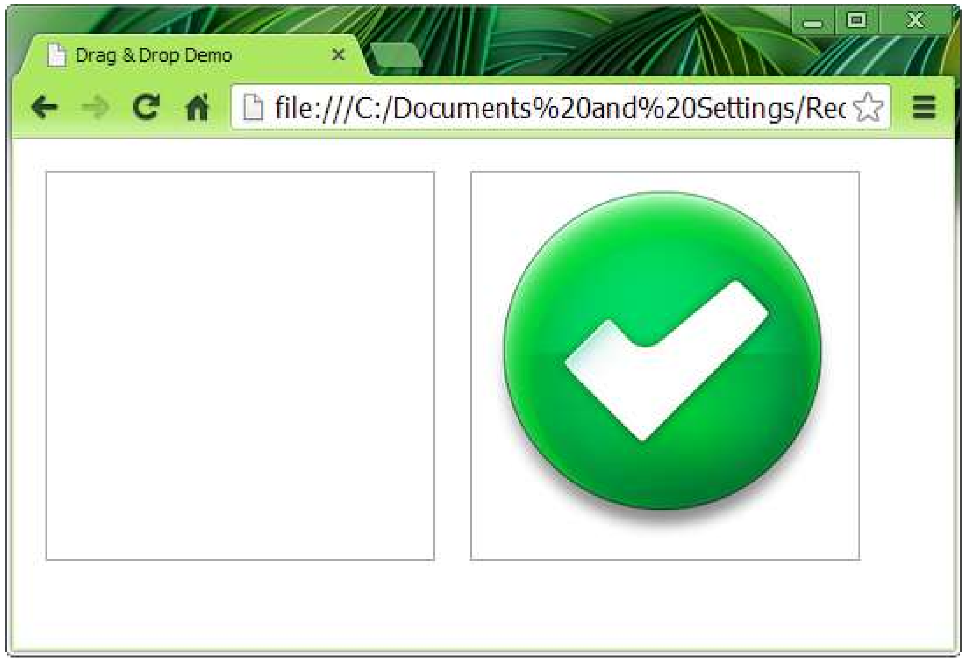
Lab2 : dragdrop1.css



```
#div1, #div2 {
    float:left; width:200px;
    height:200px;margin:10px;
    padding:10px;border:1px solid #aaaaaa;
}
```

Lab2 : Result


Drag & Drop Demo - Opera
File Edit View Bookmarks Tools Help
Drag & Drop Demo
Panels


Drag & Drop Demo
file:///C:/Documents%20and%20Settings/Rec