Trong những năm trước đây, chúng ta phải sử dụng những thư viện như JQuery và Dojo để đơn giản hóa các thành phần giao diện người dùng phức tạp như hình ảnh động, góc tròn, và kéo và thả. Giờ đây, kéo và thả đã là một phần tử chính thức trong HTML5! Nó bao gồm các sự kiện, JavaScript API, và đánh dấu bổ sung để cho phép bất kì thành phần nào trên trang web cũng có thể kéo thả được. Đặc biệt nó còn nhanh hơn, đáp ứng tốt hơn cho các ứng dụng web.

1. Kiểm tra hỗ trợ

Giống như những phần trước đã giới thiệu, khi sử dụng một API nào đó trước tiên ta cần kiểm tra xem trình duyệt hiện tại có hỗ trợ không. Thư viện tốt nhất để kiểm tra đó là “Modernizr”:

if (Modernizr.draganddrop) {

*// Browser supports HTML5 DnD.*

} else {

*// Fallback to a library solution.*

}

1. Tạo đối tượng kéo thả

Việc kéo đối tượng chỉ đơn giản là thiết lập thuộc tính draggable=true đối với các thành phần muốn làm cho nó có thể di chuyển. Bất cứ thành phần nào, miễn là nó có thể kéo đi bao gồm hình ảnh, liên kết, các tập tin, hoặc các nút DOM (Document Object Model - Mô hình Đối tượng Tài liệu).

Ví dụ:

<div id="columns">

<div class="column" draggable="true"><header>A</header></div>

<div class="column" draggable="true"><header>B</header></div>

<div class="column" draggable="true"><header>C</header></div>

</div>

1. Lắng nghe sự kiện kéo thả

Sau đây là các sự kiện trong quá trình kéo thả:

* *dragstart*
* *drag*
* *dragenter*
* *dragleave*
* *dragover*
* *drop*
* *dragend*

Đối với việc kéo thả, ta cần hiểu rõ 2 khái niệm, yếu tố nguồn – cái mà người dùng nắm kéo, ví dụ như hình ảnh, danh sách, đối tượng tập tin, liên kết,… Yếu tố đích đến – nơi mà người dùng thả đối tượng được kéo, vì vậy, không phải phần tử nào cũng có thể là đích đến.

* Dragstart

Một khi đã gắn thuộc tính draggable="true" thì kèm theo đó là xử lý ở sự kiện dragstart.

Sau đây là ví dụ làm đục đối tượng 40% khi người dùng bắt đầu kéo đối tượng:

function handleDragStart(e) {

this.style.opacity = '0.4'; *// this / e.target is the source node.*

}

var cols = document.querySelectorAll('#columns .column');

[].forEach.call(cols, function(col) {

col.addEventListener('dragstart', handleDragStart, false);

});

* Dragenter, dragover, dragleave, dragend

Sự kiện dragenter, dragover, dragleave có thể được sử dụng để xử lý đối tượng trong quá trình kéo. Ví dụ, khi một đối tượng được kéo ngang qua một đối tượng khác, hoặc khi kéo rời khỏi một đối tượng khác,…

<style>

.column.over {

border: 2px dashed #000;

}

</style>

function handleDragStart(e) {

this.style.opacity = '0.4'; *// this / e.target is the source node.*

}

function handleDragOver(e) {

if (e.preventDefault) {

e.preventDefault(); *// Necessary. Allows us to drop.*

}

e.dataTransfer.dropEffect = 'move'; *// See the section on the DataTransfer object.*

return false;

}

function handleDragEnter(e) {

*// this / e.target is the current hover target.*

this.classList.add('over');

}

function handleDragLeave(e) {

this.classList.remove('over'); *// this / e.target is previous target element.*

}

var cols = document.querySelectorAll('#columns .column');

[].forEach.call(cols, function(col) {

col.addEventListener('dragstart', handleDragStart, false);

col.addEventListener('dragenter', handleDragEnter, false);

col.addEventListener('dragover', handleDragOver, false);

col.addEventListener('dragleave', handleDragLeave, false);

});

1. Kéo tập tin

* Kéo từ máy tính vào trình duyệt

Kéo một file từ máy tính vào trình duyệt bằng cách sử dụng các sự kiện kéo thả loại khác. Sự khác biệt chính là trong việc xử lý xử lý thả. Thay vì sử dụng dataTransfer.getData() để truy cập các tập tin, dữ liệu sẽ được chứa trong thuộc tính dataTransfer.files:

function handleDrop(e) {

e.stopPropagation(); *// Stops some browsers from redirecting.*

e.preventDefault();

var files = e.dataTransfer.files;

for (var i = 0, f; f = files[i]; i++) {

*// Read the File objects in this FileList.*

}

}

* Kéo từ trình duyệt ra máy tính

Hướng dẫn: <http://www.thecssninja.com/javascript/gmail-dragout>