Leçon 7.2 – HTML5 : glisser-déposer

s1 ----------

Dans cette partie, nous allons voir comment introduire une fonctionnalité de glisser-déposer. Nous allons voir que cette fonctionnalité s'appuie sur plusieurs gestionnaires d'événement, concept que nous avons utilisé par deux fois déjà.

Grâce à plusieurs exercices de difficulté croissante, vous allez très vite comprendre comment mettre en oeuvre le glisser-déposer.

s2 ----------

Examinons quel est le principe d'un glisser-déposer et listons les éléments qui le rendent possible.

Dans une fonctionnalité de glisser-déposer, nous trouvons deux éléments :

- d'abord l'élément sur lequel on clique et qui va faire l'objet d'un déplacement : ce peut être par exemple une image, ou de façon plus large un élément de la page web ;
- ensuite la cible qui va être survolée par l'élément que l'on déplace et qui va capturer l'objet dès que la souris sera relâchée ; cette cible peut être visible ou non et plus ou moins grande.

Ce qui se passe dans un glisser-déposer du point de vue du code HTML de la page, c'est la chose suivante : l'ensemble du code HTML de l'élément déplacé - ici une image - est déplacé à l'intérieur du conteneur cible, ici un élément div.

s3 ----------

Plusieurs choses doivent être installées pour rendre possible une fonctionnalité de glisser-déposer.

Il faut d'abord d'abord rendre possible le fait qu'un élément soit déplaçable : ceci se traduit par un attribut draggable à "true" (certaines versions récentes des navigateurs – Safari, Firefox, Chrome) autorisent par défaut un comportement de glisser-déposer).

Ensuite, on s'appuie sur plusieurs événements, quatre au total, dont un est optionnel. On se rappelle que la détection d'un événement implique trois choses : un élément auquel cet événement est attaché, un gestionnaire et une fonction qui sera exécutée par ce gestionnaire.

Ainsi, concernant l'élément à déplacer, il faut détecter :

- quand il commence à être déplacé : ceci se fait grâce à un gestionnaire d'événement qui va détecter et réagir à l'événement ondragstart ;
- il peut être utile de détecter également quand l'élément a fini d'être déplacé : ceci se fait grâce à un gestionnaire d'événement qui va détecter et réagir à l'événement ondragend.

Concernant la cible, il faut détecter :

- quand elle est survolée par l'élément : ceci se fait grâce à un gestionnaire d'événement ondragover ;
- quand un élément est relâché sur cette cible : ceci se fait grâce à un gestionnaire d'événement ondrop.

Bien entendu, pour chacun de ces gestionnaires il faut définir les fonctions JavaScript qui y sont attachées. Prenons-les dans l'ordre.

La fonction associée à l'événement ondragstart est l'occasion de mémoriser à l'intérieur de l'objet JavaScript événement qui vient d'être créé l'identifiant de l'élément déplacé ; ceci se fait à travers la méthode dataTransfert.setData() (pour une explication sur l'argument "Text" de cette fonction, consulter https://developer.mozilla.org/fr/docs/DOM/DataTransfer#setData.28.29). De façon optionnelle, la fonction associée à l'événement ondragstart peut être aussi l'occasion de changer l'apparence de l'élément déplacé : un effet de transparence, par exemple.

La fonction associée à l'événement ondragend (qui est optionnel) sera pour nous l'occasion de changer à nouveau cette apparence. Ainsi, à la fin du glisser-déposer l'élément retrouvera par exemple l'aspect qu'il avait avant d'être déplacé, ou encore un autre aspect.

La fonction associée à l'événement dragover permet de spécifier où l'élément est autorisé à être déposé. Ici, la cible. Par défaut les éléments ne peuvent pas être déposés dans d'autres éléments. Ainsi la tâche essentielle de la fonction associée à l'événement ondragover est d'empêcher le fonctionnement par défaut, on va le voir à travers la méthode event.preventDefault().

Enfin, la fonction associée à l'événement ondrop réalise plusieurs choses :

- elle récupère l'identifiant de l'élément déplacé,
- et grâce à cet identifiant, transfère le code HTML de l'élément déplacé à l'intérieur du conteneur de la cible

La fonctionnalité de glisser-déposer doit être testée sur plusieurs navigateurs, notamment IE. À cet égard, une solution robuste peut être mise en place grâce à des frameworks multiplateformes comme jQuery ou jQuery UI dont nous dirons un mot plus loin.

Les exemples qui vont suivre ont été testés sur Chrome 26, Safari 6.0 et Firefox 20.

Voyons sur plusieurs exemples progressifs comment installer le glisser-déposer.

s4 ----------

Dans ce premier exemple que je vous propose de tester, nous avons une image que l'on cherche à déplacer et à laquelle est attaché un gestionnaire pour le seul événement ondragstart. Dans cet exemple, le gestionnaire est écrit à l'intérieur de la balise. Il lance une fonction fonction_ondragstart qui va ici tout simplement écrire une information dans le conteneur appelé info, et qui est vide au départ.

On note qu'aucun événement n'est pour l'instant associé au relâcher de la souris ; l'élément revient donc à sa place.

s5 ----------

Deuxième exemple : on cherche toujours à détecter le seul événement dragstart, mais le gestionnaire de cet événement est à présent défini grâce à JavaScript.

Je vous propose de tester cet exemple.

s6 ----------

Dans le troisième exemple, on détecte toujours l'événement dragstart qui nous permet par exemple de changer la transparence de l'objet déplacé et de lui ajouter un cadre. On détecte également l'événement dragend qui provoque ici un retour à la normale de la transparence de l'élément et la disparition du cadre.

Attention pour Internet Explorer, il faut utiliser filter et non opacity.

Aucun événement n'ayant été pour l'instant associé au relâcher de la souris, l'élément revient à sa place.

Je vous propose de tester cet exemple.

s7 ----------

Voyons maintenant comment mettre en place la partie "déposer" du glisser-déposer.

La première chose à faire est de modifier la fonction associée à l'événement dragstart pour sauvegarder au sein de l'objet JavaScript événement l'identité de l'élément qui est déplacé. Ceci se fait grâce à la méthode dataTransfer.

La seconde chose est d'associer à la cible (l'endroit où l'image sera déposée) un gestionnaire pour les événements ondragover et ondrop.

La fonction associée à l'événement ondragover permet donc de spécifier où l'élément est autorisé à être déposé : ici, la cible et d'empêcher le fonctionnement par défaut.

La fonction associée à l'événement ondrop permet :

- de récupérer l'identifiant de l'élément déplacé,
- et grâce à cet identifiant, de transférer l'élément déplacé à l'intérieur du conteneur de la cible.

Je vous propose de tester cet exemple désormais complet.

s8 ----------

Cette page nous donne l'occasion de voir comment l'exercice précédent pourrait être réalisé avec jQuery.

Dans ce code, avec la façon plus condensée que permet jQuery, on va retrouver les différentes fonctionnalités mises en évidence précédemment :

On rappelle qu'il est conseillé d'écrire le code JavaScript comme une fonction à l'intérieur de l'instruction document.ready, ce qui signifie que l'ensemble du code contenu à l'intérieur des accolades de document.ready ne sera exécuté qu'une fois que l'ensemble du document sera prêt.

Que fait cette fonction ?

La première chose qu'elle fait est d'associer à l'élément dont l'identifiant est image1 un gestionnaire d'événement : à chacun des deux événements dragstart et dragend, on associe une fonction ; $(this) faisant référence à l'élément sur lequel se produit l'événement. On voit que l'on exécute les mêmes choses que dans le code précédent.

La seconde chose que fait cette fonction est d'associer un gestionnaire d'événement à l'élément dont l'identifiant est cible : à chacun des deux événements dragover et drop, on associe une fonction. À nouveau, on voit que l'on exécute les mêmes choses que dans le code précédent.

Je vous propose de tester cet exemple. Bien entendu, il vous faudra importer jQuery au sein de la page HTML.

s9 ----------

Voici enfin, l'exercice que je vous propose.

Une série de billes à ranger dans deux boîtes. Par exemple les paires dans une et les impaires dans l'autre. Avec un test quand toutes les billes sont rangées.

Comme indiqué sur le code HTML joint, je vous propose d'utiliser un canvas pour chaque bille.

À vous de jouer !

Bon travail !

s10 ----------

Voici qui clôt la seconde partie de ce module consacrée à une courte introduction à certaines fonctionnalités d'HTML5. Ces fonctionnalités (dessin, animation, interactivité, glisser-déposer) qui vous permettent déjà – j'en suis sûr – d'imaginer de nombreuses animations ou jeux.

Je vous donne rendez-vous dans la troisième partie de ce module qui vous propose une méthodologie de conception et de développement d'animations HTML5.

À très bientôt.