



Par Robert DIASSÉ

Cours 2 : Les Liens en HTML - Navigation et Structure

Objectif du Cours

Comprendre l'importance des liens dans la navigation web, apprendre à créer des liens en HTML, distinguer entre les liens externes et internes, comprendre les liens ancrés, maîtriser l'attribut `href`, et connaître les types de chemins (absolu et relatif).

Introduction

Les liens jouent un rôle crucial dans la navigation web. Ils permettent de passer d'une page à une autre, de se déplacer au sein d'une même page, et d'accéder à des ressources externes. Comprendre comment créer et gérer les liens en HTML est essentiel pour la construction de sites web efficaces.

Les Fondamentaux des Liens

Les liens sur le web nous offrent la possibilité de nous déplacer entre les pages. Ils sont souvent représentés par un curseur en forme de doigt cliquable et peuvent être différenciés par leur couleur.

Il existe deux types principaux de liens :

- **Liens Externes** : Redirigent vers des pages en dehors du site.
- **Liens Internes** : Restent à l'intérieur du site.

Parmi les liens internes, on trouve les **liens ancrés**, qui permettent de naviguer à des sections spécifiques d'une même page.

Materialisation des Liens en HTML

En HTML, les liens sont créés avec la balise `<a>` (pour "anchor") et l'attribut obligatoire `href` (Hypertext Reference). La valeur de `href` peut être un chemin vers une page ou une ressource.

Types de Chemins : Absolu et Relatif

Chemin Absolu : Spécifie l'adresse complète depuis la racine du site jusqu'à la page cible. Exemple : `/dossier/page.html`.

Les chemins absolus dans un système de fichiers dépendent du système d'exploitation utilisé. Voici des exemples de chemins absolus pour différents systèmes d'exploitation :

Windows

Sur Windows, les chemins absolus commencent par la lettre de lecteur suivi de `:`. Par exemple :

- `C:\Users\nom_utilisateur\Documents\mon_fichier.txt`

Unix/Linux

Sur les systèmes Unix/Linux, les chemins absolus commencent par `/`. Par exemple :

- `/home/nom_utilisateur/documents/mon_fichier.txt`

macOS

Sur macOS, les chemins absolus sont similaires à ceux d'Unix/Linux, commençant également par `/`. Par exemple :

- `/Users/nom_utilisateur/Documents/mon_fichier.txt`

Chemin Relatif

Lorsque nous spécifions des chemins relatifs dans le développement web, nous indiquons le chemin par rapport à la page actuelle. Cette approche offre une grande flexibilité, en particulier lorsqu'il s'agit d'organiser des fichiers et des dossiers dans un projet. Voici quelques éléments clés à comprendre :

Exemple Basique

Supposons que nous ayons une structure de dossier comme suit :

```
projet/  
|-- dossier/  
|   |-- page.html  
|-- index.html
```

- Pour référencer la `page.html` étant dans `index.html`, nous utiliserions le chemin relatif : `dossier/page.html`.

Navigation Entre Dossiers parents

Utilisons `../` pour remonter d'un niveau dans la structure des dossiers.

```
projet/  
|-- dossier/  
|   |-- page.html  
|-- index.html  
|-- autreDossier/  
|   |-- autrePage.html
```

- Pour référencer la page.html étant dans autrePage.html, nous utiliserions le chemin relatif : `../dossier/page.html`.

Entrer dans un Dossier

Si nous voulons accéder à un dossier à partir du fichier actuel, nous utilisons simplement le nom du dossier suivi d'une barre oblique.

```
projet/  
|-- css/  
|   |-- style.css  
|-- index.html
```

- Pour référencer style.css étant dans index.html, nous utiliserions le chemin relatif : `css/style.css`.

Considérations de Sécurité

L'utilisation de chemins absolus peut poser des problèmes de sécurité, car cela expose la structure interne du site. C'est pourquoi les chemins relatifs sont préférés dans le développement web.

Remarques

- Les caractères de séparation de répertoire (barre oblique ou barre oblique inversée) peuvent varier en fonction du système d'exploitation. Dans le contexte web, la barre oblique (/) est généralement utilisée comme séparateur, et cela fonctionne dans la plupart des cas.
- Les chemins absolus sont couramment utilisés pour référencer des ressources externes, telles que des images ou des feuilles de style, tandis que les chemins relatifs sont privilégiés pour leur portabilité entre différents systèmes d'exploitation et configurations de serveur.

Manipulation des Liens - Exemples

1. Lien vers une Page Externe

```
<a href="https://www.google.com" target="_blank">Visiter Google</a>
```

L'attribut `target` est utilisé dans les balises `<a>` (lien hypertexte) en HTML pour spécifier où ouvrir la nouvelle page ou ressource liée par le lien. Il détermine le contexte de navigation pour la nouvelle page.

2. Lien vers une Page Interne

```
<a href="page.html">Aller à la Page</a>
```

3. Lien avec Ancre pour Déplacement sur la Même Page

Lorsque vous souhaitez créer des liens permettant de déplacer l'utilisateur vers une section spécifique de la même page, vous pouvez utiliser des liens ancrés. Cela se fait en spécifiant l'ID ou le nom de la destination comme une ancre dans le lien. Assurez-vous que la section de destination a un attribut `id` ou `name` correspondant.

Utilisation de l'ID comme Ancre

```
<!-- Dans la même page, vers une section avec l'ID "section1" -->
<a href="#section1">Aller à la Section 1</a>

<!-- Section de destination avec l'ID "section1" -->
<section id="section1">
  <h2>Section 1</h2>
  <!-- Contenu de la section -->
</section>
```

Utilisation du Nom comme Ancre (Déprécié, mais toujours pris en charge)

Dans les versions HTML plus anciennes, l'attribut `name` était utilisé pour créer des ancres. Bien que son utilisation soit dépréciée dans les versions HTML5, elle est toujours prise en charge par les navigateurs.

```
<!-- Dans la même page, vers une section avec le nom "section2" -->
<a href="#section2">Aller à la Section 2</a>

<!-- Section de destination avec le nom "section2" -->
<section name="section2">
  <h2>Section 2</h2>
  <!-- Contenu de la section -->
</section>
```

En utilisant `href="#section1"` ou `href="#section2"`, le navigateur comprend que le lien doit déplacer l'utilisateur vers la section de la même page ayant l'ID ou le nom correspondant. Le symbole `#` est utilisé pour indiquer que la destination est sur la même page. Cette technique est particulièrement utile pour créer des menus de navigation qui déplacent l'utilisateur vers différentes sections d'une page longue.

4. Lien avec Chemin Absolu

```
<!-- Lien vers une page à partir de la racine du site -->
<a href="C:\Users\DELL\Documents\page.html">Page du Dossier</a>
```

5. Lien avec Chemin Relatif

```
<!-- Lien vers une page dans le même dossier -->
<a href="autre-page.html">Autre Page</a>
```

```
<!-- Lien vers une page dans un sous dossier du nom de pages -->

<a href="pages/autre-page.html">Autre Page</a>
<!-- Lien vers une page dans un dossier parent -->

<a href="../autre-page.html">Autre Page</a>
```

La balise `<a>` (lien hypertexte) en HTML prend plusieurs attributs qui définissent son comportement, sa cible et d'autres propriétés. Voici les attributs les plus couramment utilisés avec leurs valeurs et utilisations :

1. **href** (Hypertext Reference) :

- **Valeur** : URL ou chemin vers la ressource liée.
- **Utilisation** : Définit la destination du lien. Peut être une URL externe, un chemin relatif ou un lien ancré à l'intérieur de la même page.

```
<a href="https://www.example.com">Lien Externe</a>
<a href="page.html">Lien Relatif</a>
<a href="#section1">Lien Ancré (sur la même page)</a>
```

2. **target** :

- **Valeur** : `_blank`, `_self`, `_parent`, `_top`, ou le nom d'une fenêtre ou d'un cadre.
- **Utilisation** : Définit où ouvrir la ressource liée. Par exemple, `_blank` ouvre le lien dans une nouvelle fenêtre ou un nouvel onglet.

```
<a href="https://www.example.com" target="_blank">Ouvrir dans une Nouvelle Fenêtre</a>
```

3. **download** :

- **Valeur** : Nom du fichier à télécharger.
- **Utilisation** : Indique que la cible du lien doit être téléchargée au lieu d'être naviguée.

```
<a href="document.pdf" download="mon_document.pdf">Télécharger le Document</a>
```

4. **rel** (Relations) :

- **Valeur** : Différentes valeurs, telles que `"nofollow"`, `"noopener"`, `"noreferrer"`, etc.
- **Utilisation** : Définit la relation entre le document actuel et la ressource liée.

```
<a href="https://www.example.com" rel="nofollow">Lien avec l'Attribut Nofollow</a>
```

5. **title** :

- **Valeur** : Texte descriptif.
- **Utilisation** : Fournit un texte descriptif qui apparaît généralement comme une info-bulle lorsqu'on survole le lien.

```
<a href="https://www.example.com" title="Visiter  
Example.com">Example.com</a>
```

6. **aria-* (Attributs ARIA)** :

- **Valeur** : Différentes valeurs dépendant de l'attribut ARIA spécifique.
- **Utilisation** : Améliore l'accessibilité en ajoutant des informations supplémentaires pour les technologies d'assistance.

```
<a href="page.html" aria-label="Aller à la Page">Page</a>
```

Ces attributs peuvent être combinés et utilisés en fonction des besoins spécifiques de votre lien hypertexte.

Conclusion

Les liens sont l'élément clé de la navigation web. En maîtrisant la création de liens en HTML, vous pouvez construire des sites web interactifs et bien structurés. Comprendre la différence entre les liens externes et internes, les liens ancrés, et l'utilisation des chemins absolus et relatifs vous permettra de créer des expériences de navigation fluides et sécurisées.