

Formulaires HTML

Rappel

Formulaires HTML

- <http://www.w3.org/TR/html4/interact/forms.html>
- <http://www.la-grange.net/w3c/html4.01/interact/forms.html>
- http://www.w3schools.com/html/html_forms.asp
- En HTML un formulaire recueille des informations de l'utilisateur, puis les transfère au serveur, qui à son tour doit les traiter
- Un formulaire débute par la balise `<form>` (et finit donc par `</form>`). Ce conteneur contient toutes les autres balises du formulaire. Il peut contenir aussi d'autres balises : `<p>`, `<table>`, etc., permettant la mise en forme (présentation) du formulaire

Exemple

- `<form >`
- `<label for="prenom">Prénom : </label>`
- `<input type="text" id="prenom" ><br>`
- `<label for="nom">Nom : </label>`
- `<input type="text" id="nom"><br>`
- `<label for="email">e-mail: </label>`
- `<input type="text" id="email"><br>`
- `<input type="radio" name="sexe" value="homme">`
homme

- `<input type="radio" name="sexe" value="femme">`
femme

- `<input type="submit" value="envoyer"> <input`
type="reset" >
- `</form>`

Prénom :

Nom :

E-mail:

☐ Homme
☐ Femme

La balise `< input >`

- Sert à spécifier comment s'effectue la saisie du formulaire : par cases à cocher, boutons radio, texte libre, etc.
- L'attribut « *type* » permet de préciser s'il s'agit de cases à cocher, boutons radio, etc.
- Les valeurs de l'attribut « *type* » : “button”, “checkbox”, “file”, “hidden”, “image”, “password”, “radio”, “reset”, “submit”, “text”

La balise `<input>` - submit

- Bouton pour l'envoi des données
- Exemple :
 - Nom : `<input name="nom" /><br />`
 - E-mail : `<input name="email" /><br />`
 - `<input type="submit" value="Envoyer" />`

- Résultat :



Nom :

E-mail :

La balise `< input >` - image

- Bouton pour la soumission de données représenté par une image
- Exemple :
 - `<input type="image" src="imageBouton.gif" alt="Submit" />`
- Attributs :
 - `src` – pour préciser l'image à afficher
 - `alt` – texte alternatif

La balise `<input>` - reset

- Réinitialisation du formulaire
- Exemple :
 - `<input type="text" />`
 - `<input type="submit" />`
 - `<input type="reset" />`
- Résultat :



A visual representation of the HTML code's output. It shows a white rectangular text input field on the left. To its right are two gray rectangular buttons with rounded corners. The first button contains the text 'Envoyer' and the second button contains the text 'Effacer'.

La balise **< input >** - file

- Bouton pour charger un fichier
- Exemple :
 - `<input type="file" />`
- Attributs (optionnels) :
 - name- pour préciser le nom du bouton
 - accept - pour préciser les types de fichiers acceptés (MIME type)
 - Exemple : `<input type="file" name="image" id="image" accept="image/gif, image/jpeg" />`

La balise `<textarea>`

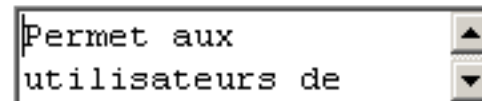
- Permet aux utilisateurs de laisser un commentaire ou de taper un texte quelconque

- Exemple :

```
<textarea rows="2" cols="20">
```

Permet aux utilisateurs de laisser un commentaire ou de taper un texte quelconque...

```
</textarea>
```



- Attributs :

- name : nom de l'élément
- rows et cols : nombre de lignes et de colonnes de la zone de texte - obligatoires. Valeur numérique.
- readonly : la modification du contenu n'est pas permise. Valeur : « readonly »
- disabled : désactive l'élément. Valeur : « disabled »
- Le texte entre les balises `<textarea>` et `</textarea>` est le texte par défaut

Les balises `<select>` et `<option>`

- Affiche une liste d'options parmi lesquelles l'utilisateur doit choisir. Le choix peut être unique ou multiple

- Exemple :

`<select>`

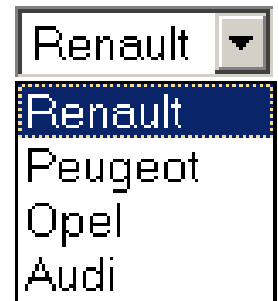
`<option value = "renault">Renault</option>`

`<option value = "peugeot">Peugeot</option>`

`<option value = "opel">Opel</option>`

`<option value = "audi">Audi</option>`

`</select>`



Les balises `<select>` et `<option>`

- Attributs `<select>` :
 - name : nom de la liste `<select>`
 - size : le nombre d'options visibles (size="1" - liste déroulante)
 - multiple : permet à plusieurs options d'être sélectionnées simultanément. Valeur : « multiple ».
 - disabled : désactive la liste. Valeur : « disabled »
- Attributs `<option>` :
 - label : étiquette (courte) de l'option
 - value : la valeur qui sera envoyée au serveur
 - selected : précise si l'option est sélectionnée par défaut. Valeur : « selected ».
 - disabled : désactive l'option. Valeur : « disabled »

La balise <optgroup>

- Permet de regrouper les éléments d'une liste d'options

- Exemple :

```
<select>  
  <optgroup label="Voitures françaises">  
    <option value="renault">Renault</option>  
    <option value="peugeot">Peugeot</option>  
  </optgroup>  
  <optgroup label="Voitures allemandes">  
    <option value="mercedes">Mercedes</option>  
    <option value="audi">Audi</option>  
  </optgroup>  
</select>
```



- Attributs :

- label : étiquette du groupe d'options. Attribut obligatoire.
- disabled : désactive le groupe d'options. Valeur : « disabled »

La balise <label>

- Permet d'associer à un élément de formulaire des informations relatives à son contenu
- Exemple :

```
<label for="nom">Nom : </label>  
<input type="text" id="nom">
```
- Attribut(s) :
 - for : renvoie à l'élément de formulaire spécifié par le même attribut id

La balise <fieldset>

- Utilisée pour regrouper les éléments d'un formulaire
- Exemple :
- <fieldset >
 - <legend>Données personnelles :</legend>
 - <label for="nom">Nom : </label>
 - <input type="text" size="20" id="nom" />

 - <label for="prenom">Prénom : </label>
 - <input type="text" size="20" id="prenom" />

 - <label for="date"> Date de naissance : </label>
 - <input type="text" size="10" id="date" />
- </fieldset>
- Résultat :

Données personnelles :

Nom :

Prénom :

Date de naissance :

La balise <button>

- Affiche un bouton. Il est possible de placer du contenu (texte ou image sur le bouton)
- Exemple :
- `<button type="button">Test</button>`
- Attributs :
 - type : le type du bouton : « *submit* », « *reset* » ou « *button* »
 - value : la valeur qui sera envoyée au serveur
 - name : le nom du bouton
 - disabled : désactive le bouton. Valeur : « *disabled* »

La balise `<input>` - text

- Pour saisir une ligne de texte

- Exemple :

Nom : `<input type="text" id="nom" />`

- Résultat :

Nom :

- Attributs :

- value : valeur initiale (par défaut)
- size : taille (dans la plupart des navigateurs, la taille par défaut est de 20 caractères)
- maxlength : le nombre maximal de caractères que l'utilisateur peut saisir
- readonly : le contenu ne peut pas être modifié

La balise `<input>` - checkbox

- Case à cocher (checkbox)
- Exemple :
 - Marque voiture : `<br />`
 - Renault `<input type="checkbox" name="vehicule" value="Renault" /><br />`
 - Peugeot `<input type="checkbox" name="vehicule" value="Peugeot" /><br />`

- Résultat :

```
Marque voiture :  
Renault ☐  
Peugeot ☐
```

- Attribut :

- `checked = "checked"`
défaut

précise si la case est cochée par

La balise `<input>` - radio

- Outre la présentation (les cases sont rondes et non plus carrées), la différence entre les cases à cocher et les boutons radio : les boutons radio qui portent le même *name* s'excluent mutuellement : cocher un $\Rightarrow$ décocher les autres.

- Exemple :

- `<input type="radio" name="sexe" value="homme" /> Homme<br />`
- `<input type="radio" name="sexe" value="femme" /> Femme<br />`

- Résultat :

- Attribut :
 - `checked="checked"` :  Homme  Femme bouton est sélectionné par défaut

La balise `<input>` - password

- A l'identique que le type text, mais les caractères tapés sont affichés sous forme d'astérisques
- Attention : le mot de passe est envoyé en clair dans le message final. Aucune méthode de cryptage n'est appliquée, la confidentialité ne se limite qu'à l'instant de la saisie

- Exemple :

- Nom : `<input type="text" id="nom" /><br>`
- Mot de passe : `<input type="password" id="motDePasse" />`

- Résultat :

Nom :

Mot de passe :

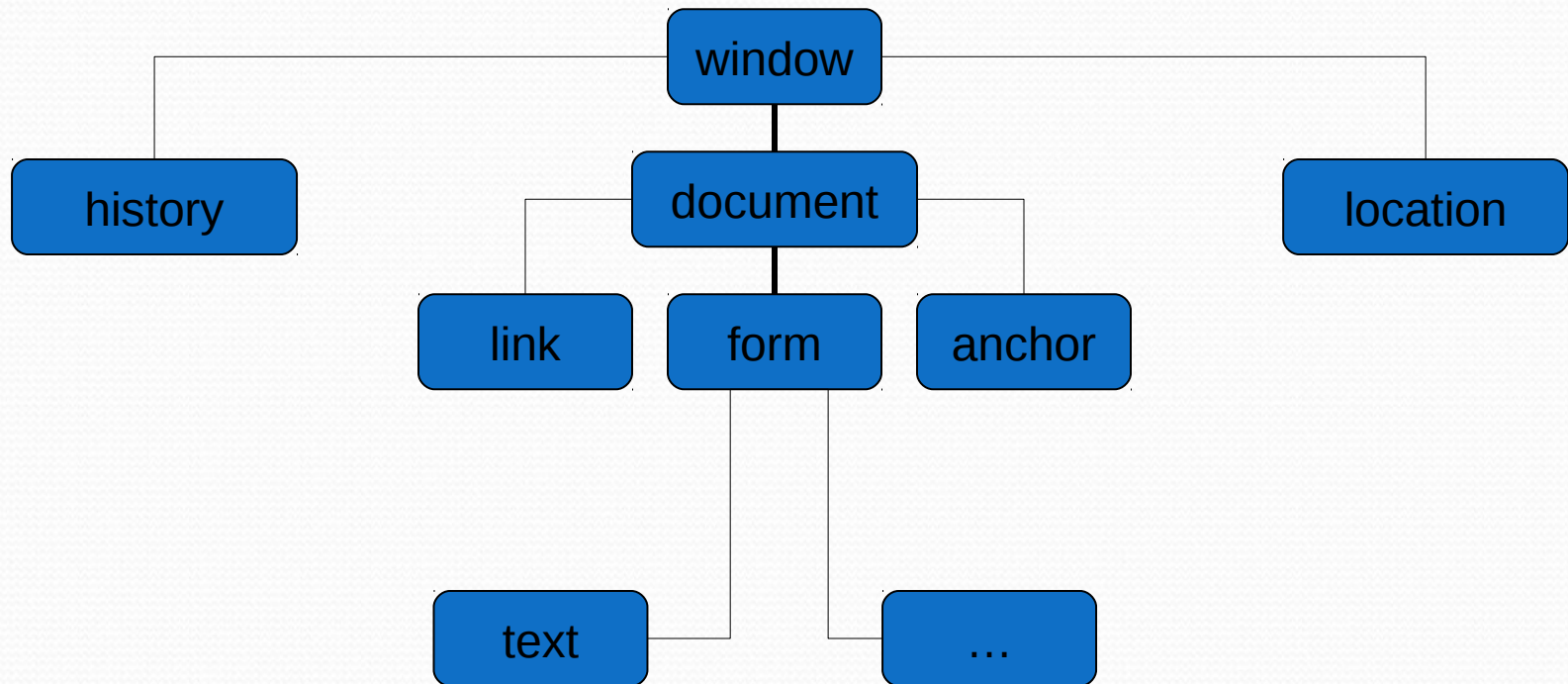
- Attributs : comme pour le text

JavaScript, événements et DOM

Modèle objet et JavaScript

- Chaque élément (balise) HTML devient un objet hôte fourni par le navigateur (qui satisfait la description du DOM – Document Object Model)
- Attention ! Chaque navigateur a son implémentation du DOM
- Hiérarchie d'objets

Modèle objet



L'objet *document*

- Représente la page HTML et donne accès à tous ses éléments.
- Ensembles d'objets :

Collection	Description
anchors[]	Retourne une reference vers tous les objets Anchor dans le document
applets[]	Retourne une reference vers tous les objets Applet dans le document
forms[]	Retourne une reference vers tous les objets Form dans le document
images[]	Retourne une reference vers tous les objets Image dans le document
links[]	Retourne une reference vers tous les objets Lien (hypertexte) dans le document

L'objet « document »

- Méthodes

Méthode	Description
getElementById()	Retourne l'adresse du premier objet qui a l'identifiant (id) spécifié
getElementsByName()	Retourne une collection d'objets qui ont le nom (name) spécifié
getElementsByTagName()	Retourne une collection d'objets qui ont la balise spécifiée
write()	Ecrit une chaîne de caractères dans un document

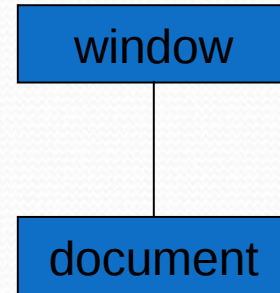
- Attention : la liste de méthodes n'est pas complète, ni la liste des objets DOM ! Pour plus de détails voir http://www.w3schools.com/html/dom/dom_intro.asp

Modification du contenu

- Ajout d'un élément :
 - **createElement** permet d'ajouter une balise avec comme argument le nom de la balise souhaitée
 - Exemple : `nouveaudiv=document.createElement("div")`
 - **createTextNode** permet d'ajouter du texte
 - Exemple : `montexte=document.createTextNode("Bonjour")`
 - **cloneNode(deep)** permet de créer une copie de l'élément courant. Le paramètre **deep** permet de paramétrer le niveau de profondeur de la copie
 - `deep = false` ⇒ seul le nom de la balise, ainsi que ses attributs sont copiés
 - `deep = true` ⇒ le nom de la balise, les attributs et le contenu sont copiés
 - Le résultat de ces méthodes est un nœud n'ayant aucun parent. Il faut ensuite l'affecter à un nœud déjà existant en utilisant la méthode **appendChild()**
 - Exemple : `elementParent.appendChild(elementFils)`
- Supprimer un nœud
 - **RemoveChild**
- Modifier un attribut
 - **setAttribute**

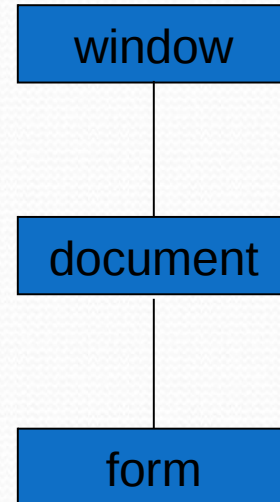
Example

- `<html>`
- `<body>`
- `<h1> Titre`
`</h1>`
- `</body>`
- `</html>`



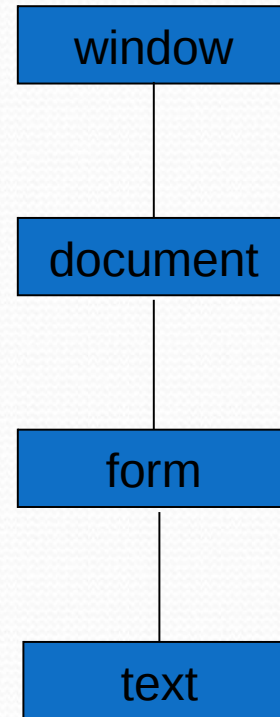
Example

- `<html>`
- `<body>`
- `<h1> Titre </h1>`
- `<form>`
- `</form>`
- `</body>`
- `</html>`



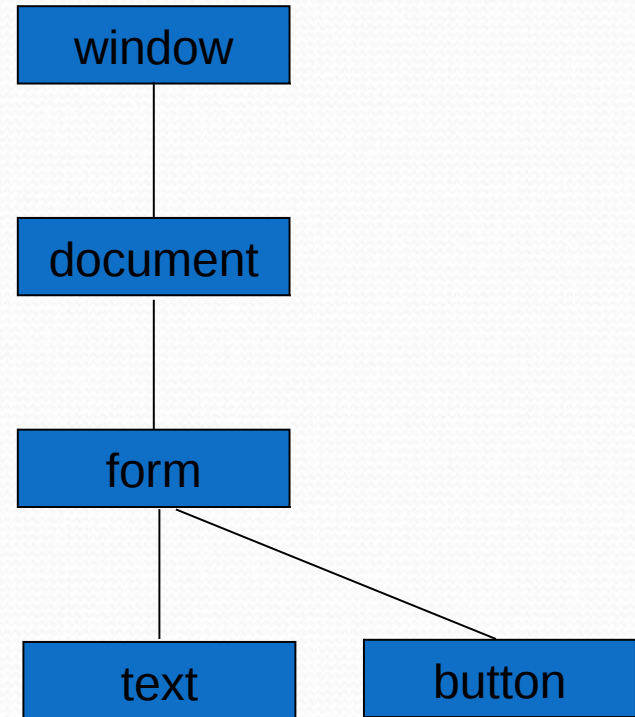
Example

- `<html>`
- `<body>`
- `<h1> Titre </h1>`
- `<form>`
- `<input type="text">`
- `</form>`
- `</body>`
- `</html>`



Example

- `<html>`
- `<body>`
- `<h1> Titre </h1>`
- `<form>`
- `<input type="text">`
- `<input type="button">`
- `</form>`
- `</body>`
- `</html>`



Appeler un objet

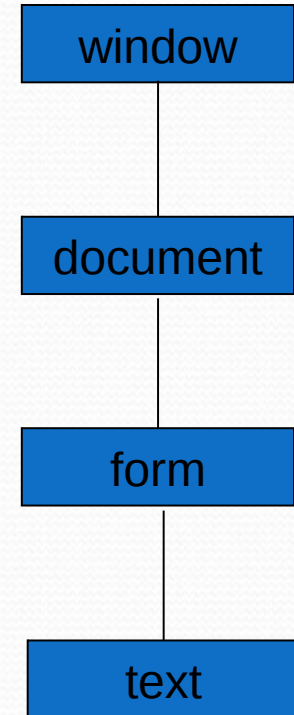
- Chaque élément HTML peut avoir un nom
 - attribut " name " ou " id " - à partir de HTML 4.0
- Exemple :
- `<form id = "formulaire">`
- `<input type="text" id = "donnee">`
- `</form>`
- Règles pour les noms :
 - ne peuvent pas contenir d'espace, ni de caractères de ponctuation (excepté " _ ")
 - doivent commencer par une lettre
 - doivent être entourés de ""

Exemple

- `<html>`
- `<body>`
- `<form name = "formulaire">`
- `<input type="text" id = "donnees">`
- `</form>`
- `</body>`
- `</html>`

- Utilisation JavaScript :

`window.document.formulaire.donnees`



Propriétés

- Les attributs HTML définissent les propriétés des objets JavaScript
- `<form id = " formulaire ">`
- `<input type = " text " id = "nom" value=" Nom utilisateur " >`
- `</form>`
- `document.formulaire.nom.value`

Evénements

- Les événements sont des actions qui peuvent être détectées par JavaScript.
- Chaque balise d'une page Web peut contenir certains événements qui peuvent déclencher des méthodes/fonctions de Javascript
- Exemples d'événements :
 - Un clic de souris
 - Un chargement de page Web ou d'une d'image
 - Sélection d'un champ INPUT dans un formulaire HTML
 - Soumission d'une forme de HTML
 - Appuyer sur une touche
- Dans l'élément HTML il faut préciser un attribut supplémentaire :
 - *nomEvénement* = " *instructions* "
- Exemple :
 - `<input type = " button " value = " Clic " onClick = " window.alert(' Clic !!') ">`

Fonctions

- Une fonction est un bloc réutilisable de code (défini dans la section head) qui sera exécuté par un événement, ou à l'appel de la fonction.

- Exemple :

```
<html>
<head>
  <script type="text/javascript">
    function displaymessage()
    { alert("Bonjour!") }
  </script>
</head>
<body>
  <form>
    <input type="button" value="Test" onclick="displaymessage()" >
  </form>
</body>
</html>
```

définition fonction

événement

appel fonction

Opérations sur les formulaires

Accès à un formulaire

- Avec un numéro d'index
 - Exemple : `document.forms[index]`
 - Le premier formulaire avec `forms[0]`, le deuxième avec `forms[1]` etc... dans l'ordre dans lequel les balises `<form>` apparaissent dans le fichier
- Avec le nom du formulaire comme index
 - Exemple : `document.forms["nomFormulaire"]`
 - où *nomFormulaire* est la valeur de l'attribut `name=` dans la balise `<form>`
- Directement avec le nom du formulaire
 - Exemple : `document.nomFormulaire`
 - où *nomFormulaire* est la valeur de l'attribut **name** dans la balise `<form>`

Accès aux éléments d'un formulaire

- Pour accéder aux éléments du formulaire (à des champs de saisie, des boutons etc.) utilisez l'objet ***elements***
- Avec un numéro d'index
 - Exemple : `document.forms[indexForm].elements[indexElement]`
- Avec le nom de l'élément
 - Exemple : `document.nomFormulaire.nomElement`
 - où *nomFormulaire* et *nomElement* sont les noms du formulaire et de l'élément dans les balises HTML correspondants (attribut ***name***)

Exercice 1

- Pour le formulaire suivant, si l'utilisateur coche le bouton « Femme » ainsi que la case « Marié(e) », donnez accès au champ de saisie « Nom de jeune fille » qui par défaut est caché

Nom :

Prénom :

Marié(e) ☐

Sexe : ☐ Homme ☐ Femme

Nom de jeune fille :

- Indications :
 - pour les boutons radio et checkbox utilisez l'attribut « checked » (valeurs true ou false)
 - les boutons radios sont stockés dans un tableau du nom des boutons radio
 - Utilisez `setAttribute("attribute","nom")` pour passer d'un type hidden à un type texte

Vérification de la saisie et chaînes de caractères

- Objet « ***String*** »
- Exemples :
 - var valeur = 'test';
 - var chaine = ***new*** String(valeur) ;
 - var chaine1 = "Bonjour !"
- Propriété de l'objet String : ***length***
- Exemples :
 - longueur = chaine.length;
 - longueur1 = ("Bonjour !").length;

Méthodes de l'objet String

(liste non-exhaustive !)

- **toLowerCase**(chaîne) - convertit la chaîne en minuscules
- **toUpperCase**(chaîne) - convertit la chaîne en majuscules
- **charAt**(position) - retourne le caractère qui se trouve à la position spécifiée
- **charCodeAt**(position) - retourne un nombre qui représente la valeur du code Unicode du caractère qui se trouve à la position spécifiée
- **indexOf**(valeur, debut) - recherche une sous-chaîne de caractères dans l'objet String, à partir de la position *debut*
- **lastIndexOf**(valeur, debut) - recherche de la dernière apparition d'une sous-chaîne de caractères
- **substr**(debut, longueur) - retourne une sous-chaîne de caractères
- **substring**(debut, fin) - retourne une sous-chaîne de caractères

- Utilisation : *objet.methode()*
- Exemple : *chaîne.toUpperCase()*

Exercice 2

- Ecrivez une fonction `verifier_email()` qui est appelée suite à un changement dans un champ de texte et qui vérifie que le texte contient le symbole "@". Affichez un message (alert) qui dit si l'adresse est bonne ou non.
- `<input type="text" size="30" id="email" onchange=" verifier_e_mail()">;`
- Indication : pour récupérer la valeur du champ utilisez `document.nom_forme.email.value`

Exercice 3

- Ajoutez deux champs de texte Nom et Prénom et vérifiez que l'adresse e-mail est bien `nom.prénom@univ-lyon2.fr`

Exercice 4

- Réalisez un formulaire qui sert de calculatrice (supportant des additions et soustractions)