

## Introduction

Les formulaires occupent une place prépondérante dans la conception et l'exploitation d'une application ou un site Web. C'est en effet la seule façon de recevoir en retour des informations provenant directement de l'utilisateur final. Il suffit de penser à tous les sites à vocation commerciale pour lesquels ces formulaires sont indispensables.

La préoccupation première des développeurs d'application Web est de récupérer des données valides. Citons par exemple un code postal français avec cinq caractères numériques ou une adresse email dont la syntaxe est valide. Une des applications premières du JavaScript a été (et est toujours) la validation en direct (côté client) des données encodées par l'utilisateur. Une des idées majeures du Html5 est de faire prendre en charge par le Html (et donc par les navigateurs) la validation des champs de formulaires, libérant ainsi le code source de scripts redondants. Et ce, même si JavaScript est désactivé dans le navigateur.

Les formulaires font partie du langage Html depuis une quinzaine d'années et n'ont pas été modifiés depuis. En outre avec le Web 2.0, l'utilisateur s'est habitué à une présentation plus proche des applications logicielles que des pages Web classiques. Citons les calendriers qui sont devenus usuels dans les sites de réservation en ligne ou les curseurs utilisés dans divers logiciels.

En intégrant dans le langage Html les spécifications Xforms, le Html5 marque indéniablement les esprits en ajoutant aux formulaires hérités du Html 4.0 un nombre impressionnant de formulaires inédits, se rapprochant de ce qui est utilisé dans les applications logicielles, mais malheureusement non pris en charge par tous les navigateurs.

Le but ultime des formulaires est le traitement automatisé des données récoltées. Il faudra alors faire appel, côté serveur, à des langages de programmation et de gestion de bases de données comme par exemple PHP et MySQL.

A la fin de cette présentation vous serez capable de concevoir une interface contenant le formulaire suivant :

*Client*

Nom et prénom :

Adresse:

Code Postal :

Pays :

Message pour la commande :

Votre message ici...

*Commande*

Modèle :

Taille :

S

M

L

XL

Quantité :

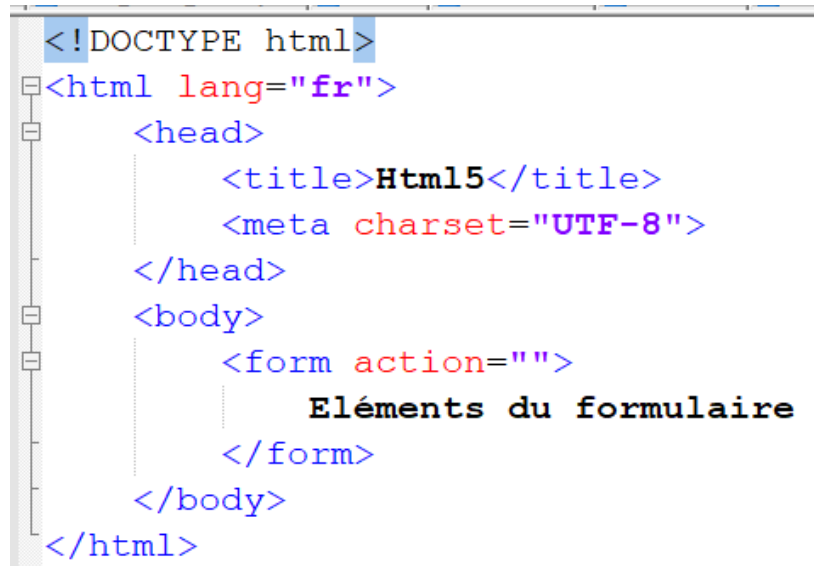
Commander

Recommencer

Afficher la commande

## 1) La déclaration d'un formulaire

Ce sont les balises **<form>** ... **</form>** qui permettent de prévenir le navigateur qu'un formulaire est créé :



```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <title>Html5</title>
    <meta charset="UTF-8">
  </head>
  <body>
    <form action="">
      Eléments du formulaire
    </form>
  </body>
</html>
```

Les attributs usuels de la balise **<form>** sont :

- **name** Pour attribuer un nom au formulaire.
- **action** Lorsque vous donnez l'ordre au navigateur de transmettre les données du formulaire, il a besoin de connaître l'action qu'il doit effectuer (Si vous utilisez le validateur du W3C <http://validator.w3.org> , la présence de l'attribut action est obligatoire). Cette action sera :
  - soit l'adresse d'un programme de traitement des données, situé sur le serveur, en CGI, Perl, PHP, ASP... Par exemple, **action = "http://www.serveur/traitement.php"**.
  - soit une adresse de courrier électronique pour récupérer simplement les données. Le protocole mailto est alors utilisé. Par exemple, **action="mailto:mon\_email@serveur"**.
  - soit, lorsque les données d'un formulaire sont traitées en interne (côté client) par du JavaScript, l'attribut **action** reste vide. Par exemple **action=""**
- **enctype** L'attribut enctype spécifie sous quel format informatique (mime type) seront transmises les données du formulaire. Par défaut, la valeur est application/x-www-form-urlencoded. Pour l'envoi de fichiers (voir la section "Les formulaires de transfert de fichiers" dans ce chapitre), celle-ci doit être multipart/form-data et enfin, pour l'envoi vers une adresse électronique avec le protocole mailto, la valeur sera text/plain.
- **method** La transmission des données d'un formulaire s'effectue selon la méthode GET ou la méthode POST. La méthode GET effectue le transfert en caractères ASCII et les données ne peuvent excéder 100 caractères. La méthode POST gère les caractères non ASCII et une quantité de caractères illimitée. Dans la pratique, la méthode POST s'est imposée pour des raisons de facilité, d'efficacité et de sécurité.

## 2) Les différentes zones de texte

- Ligne de texte

Permet à l'utilisateur de saisir du texte (alphabétique et/ou numérique) sur une seule ligne.

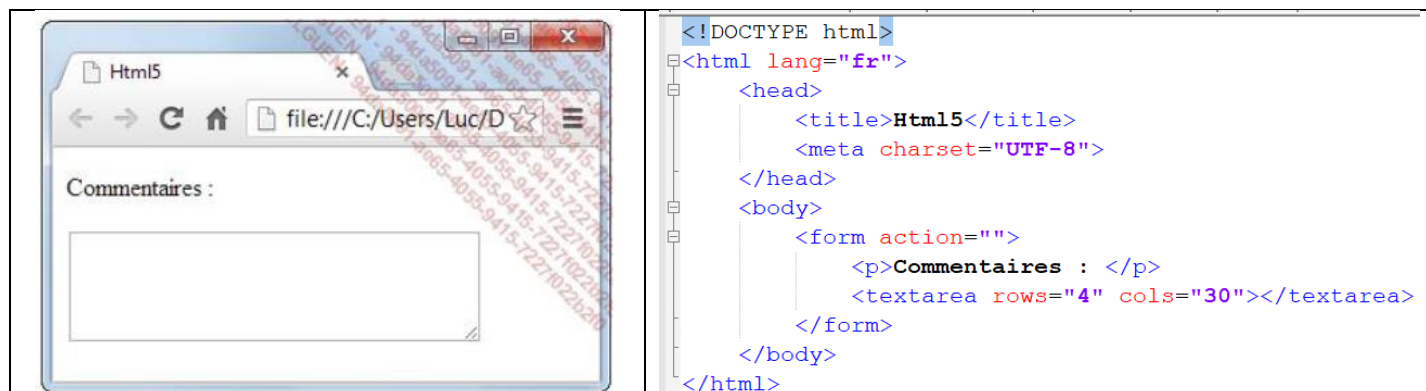


Les attributs possibles :

- |               |  |
|---------------|--|
| ○ name        | définit un nom unique dans le formulaire (pour le PHP)             |
| ○ id          | définit un nom unique dans le formulaire (pour CSS et JavaScript)  |
| ○ size        | nombre de caractères visibles                                      |
| ○ maxlength   | nombre de caractères maximum que l'utilisateur peut saisir         |
| ○ value       | valeur par défaut  |
| ○ readonly    | indique que la valeur par défaut ne pourra pas être modifiée       |
| ○ placeholder | affiche en grisé une suggestion, mais ne constitue pas une valeur  |
| ○ autofocus   | donne le focus au chargement de la page                            |
| ○ required    | rend la saisie de la zone obligatoire                              |
| ○ pattern     | permet de vérifier le contenu par une expression régulière (regex) |
| ○ height      | précise la hauteur de la ligne de texte (en pixel ou en %)         |
| ○ width       | précise la largeur de la ligne de texte                            |

- Zone de texte

Permet à l'utilisateur de saisir du texte (alphabétique et/ou numérique) plusieurs lignes.

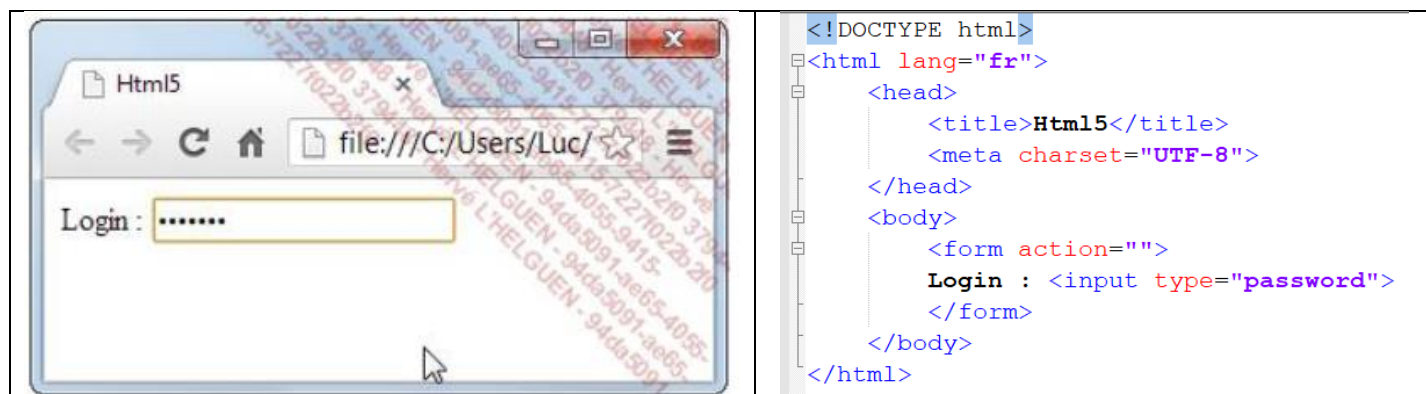


Les attributs possibles :

- name définit un nom unique dans le formulaire (pour le PHP)
- id définit un nom unique dans le formulaire (pour CSS et JavaScript)
- cols nombre de caractères visible en largeur
- rows nombre de lignes visibles en hauteur
- readonly indique que la valeur par défaut ne pourra pas être modifiée
- placeholder affiche en grisé une suggestion, mais ne constitue pas une valeur
- autofocus donne le focus au chargement de la page
- required rend la saisie de la zone obligatoire
- wrap indique si les retours à la ligne sont transmis **wrap = "hard"** ou pas (valeur par défaut) **wrap = "soft"**

- Ligne de mot de passe

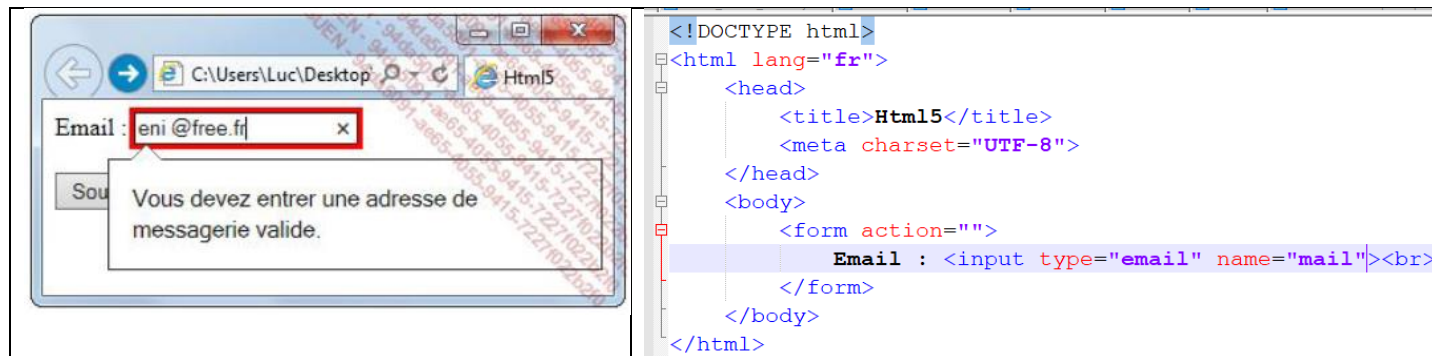
Permet à l'utilisateur de saisir un mot de passe (alphabétique et/ou numérique) sur une seule ligne, chaque caractère saisi est remplacé par une puce. Attention, ceci ne protège en rien le mot de passe qui sera transmis en clair !



Les attributs possibles sont ceux d'une ligne de texte.

- Zone e-mail

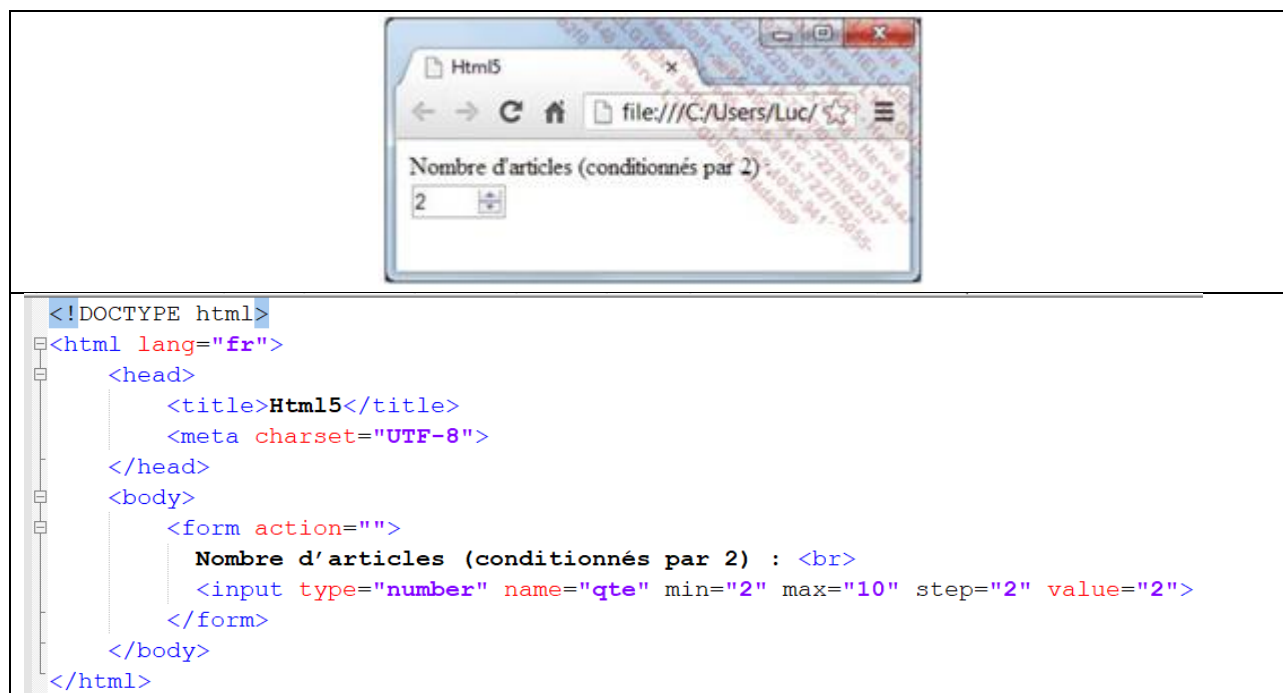
Permet à l'utilisateur de saisir une adresse mail sur une seule ligne. Le navigateur vérifie que l'adresse saisie possède un format valide.



Les attributs possibles sont ceux d'une ligne de texte.

- Zone de texte numérique

Permet de choisir une valeur numérique



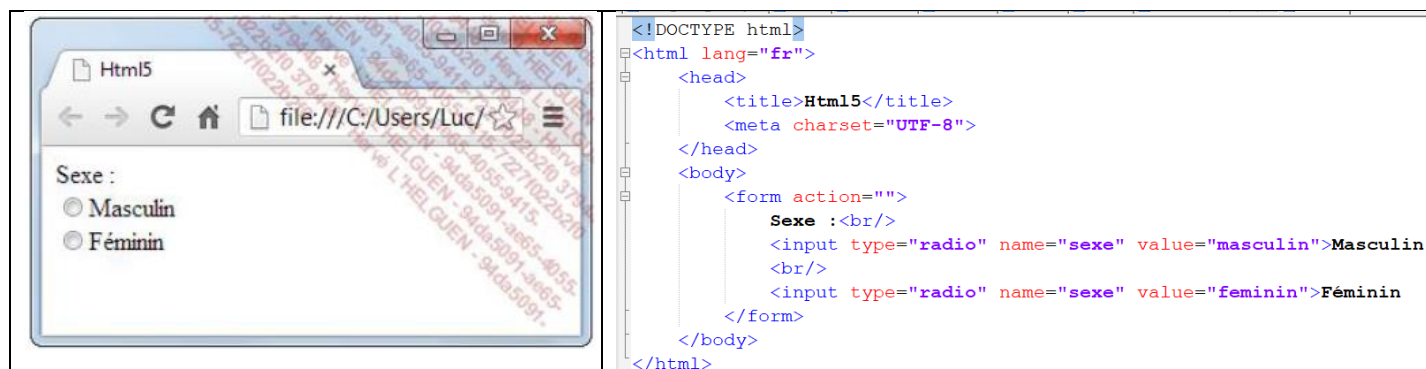
Les attributs possibles :

- min Indique la valeur minimale
- max Indique la valeur maximale
- step Indique le pas d'avancement
- value Indique la valeur de départ du compteur

### 3) Bouton radio et case à cocher

- Bouton radio

Permet à l'utilisateur de choisir une option parmi toutes celles proposées.



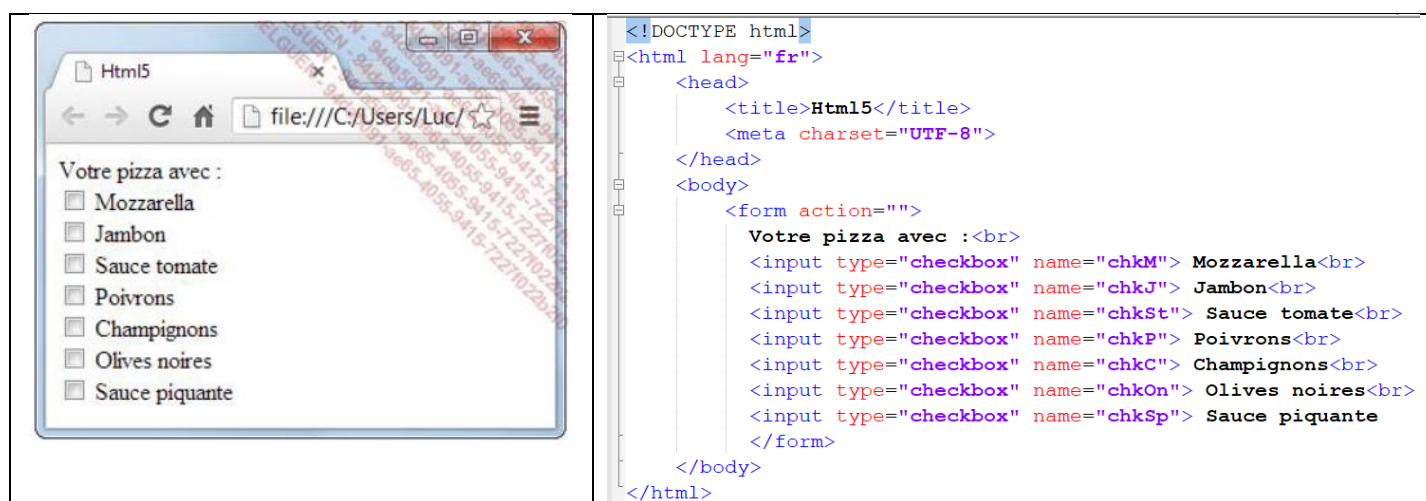
Vous remarquerez que les 2 boutons portent le même nom (puisque seul sera pris en compte celui qui est coché) !

Les attributs possibles :

- name définit un nom unique dans le formulaire
- checked permet de présélectionner un bouton radio
- value valeur qui sera exploitée lors de l'envoi du formulaire

- Case à cocher

Permet à l'utilisateur de choisir aucune, une ou plusieurs options parmi toutes celles proposées.

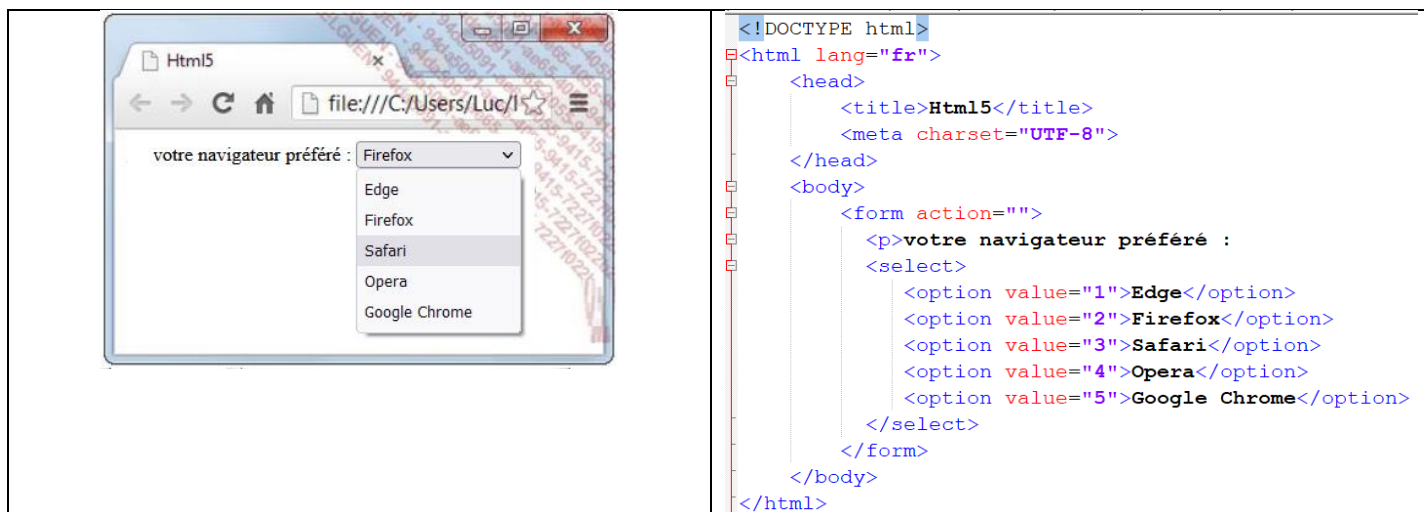


Les attributs possibles sont les mêmes que pour le bouton radio.



#### 4) Liste déroulante

Permet à l'utilisateur de choisir un élément parmi une liste.

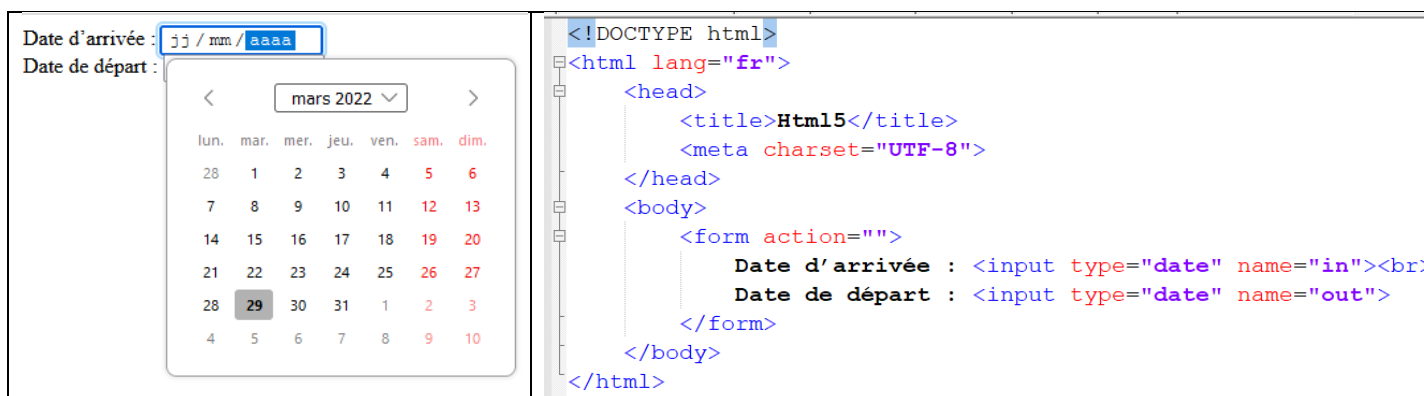


Les attributs possibles :

- name définit un nom unique dans le formulaire
- size définit le nombre d'éléments visibles (peu d'intérêt)

#### 5) Calendrier

Permet de choisir une date et une heure avec la possibilité de choisir le format adapté.



L'attribut type peut prendre les valeurs suivantes :

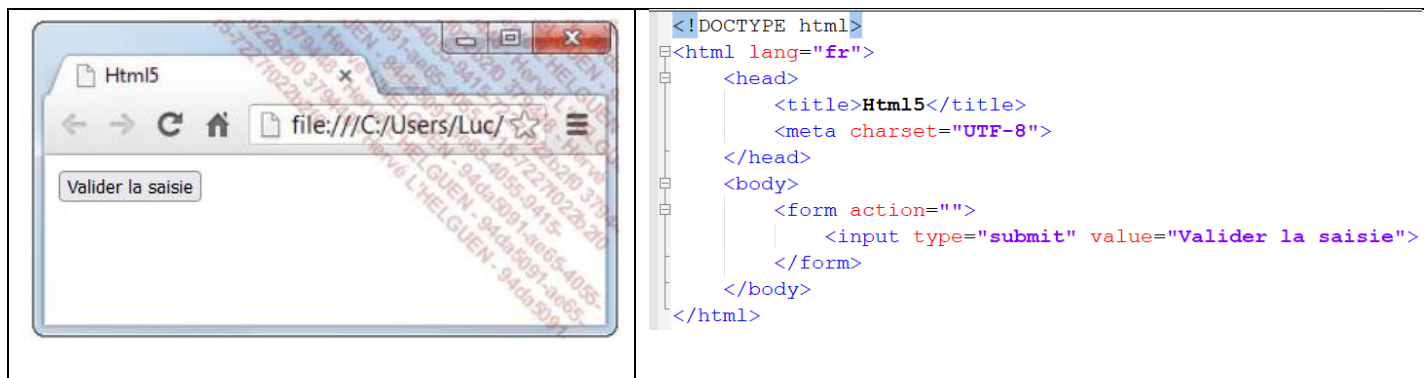
- date
- time
- datetime
- month
- week

Attention, le rendu de ce composant peut varier d'un navigateur à un autre.

## 6) Les boutons

- Bouton d'envoi (submit)

Permet à l'utilisateur de valider et d'envoyer les éléments saisis dans le formulaire. Suite à cet évènement, la commande précisée dans l'attribut action du formulaire est exécuté.



- Bouton d'annulation

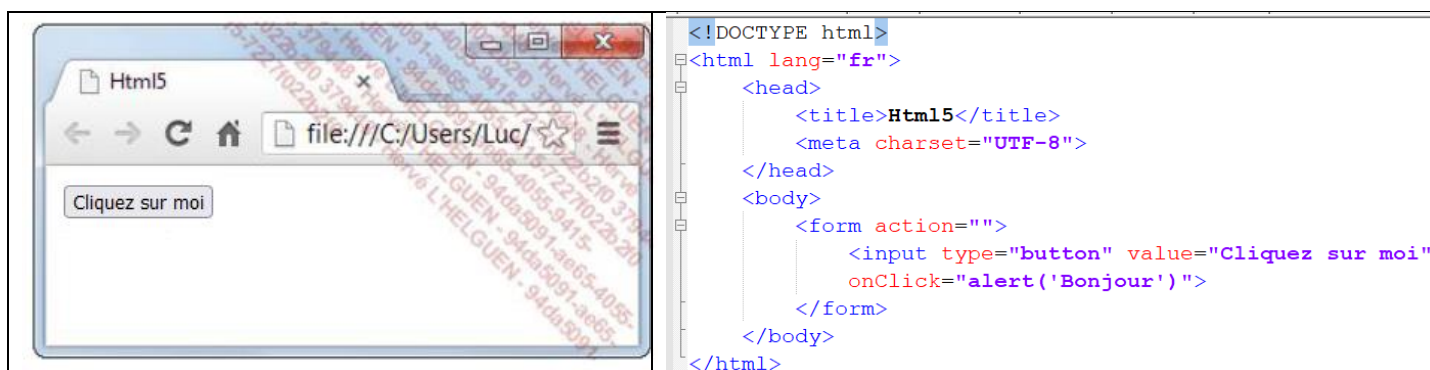
Permet de réinitialiser tous les composants du formulaire.



Remarque : le texte affiché dépend du navigateur (Effacer, Réinitialiser, ...).

- Bouton de commande

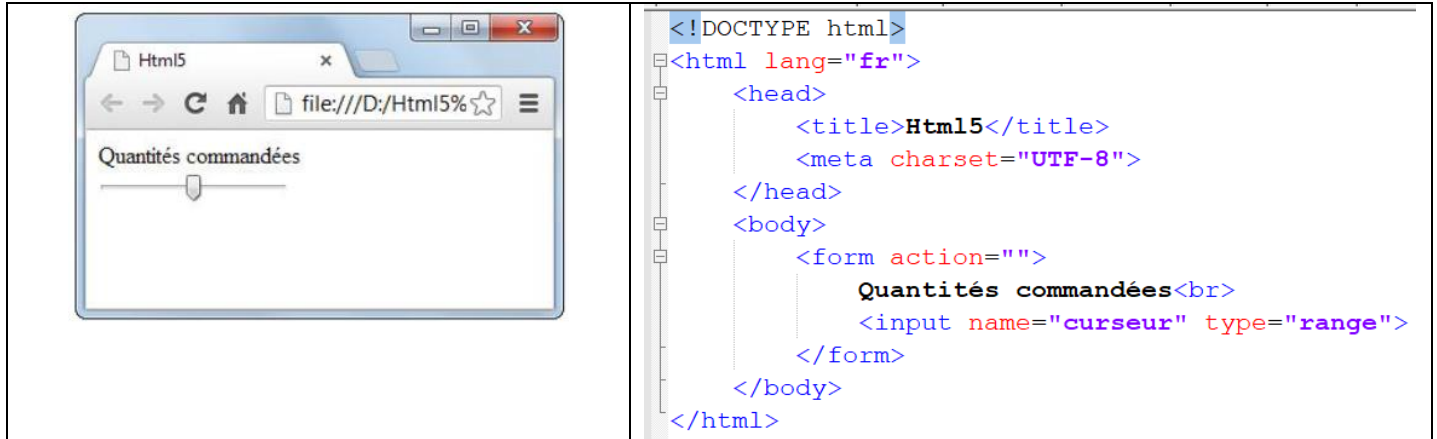
Permet d'exécuter un script (par exemple du code JavaScript)





## 7) Curseur

Permet de fixer une valeur numérique de façon conviviale.



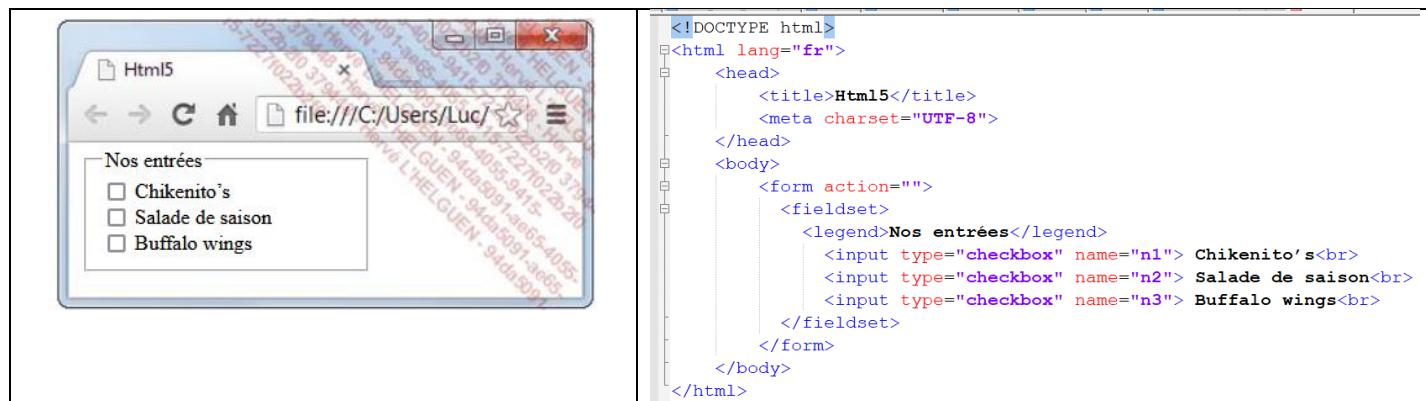
Les attributs possibles :

- min Indique la valeur minimale
- max Indique la valeur maximale
- step Indique le pas d'avancement
- value Indique la valeur de départ du compteur

## 8) Organisation du formulaire

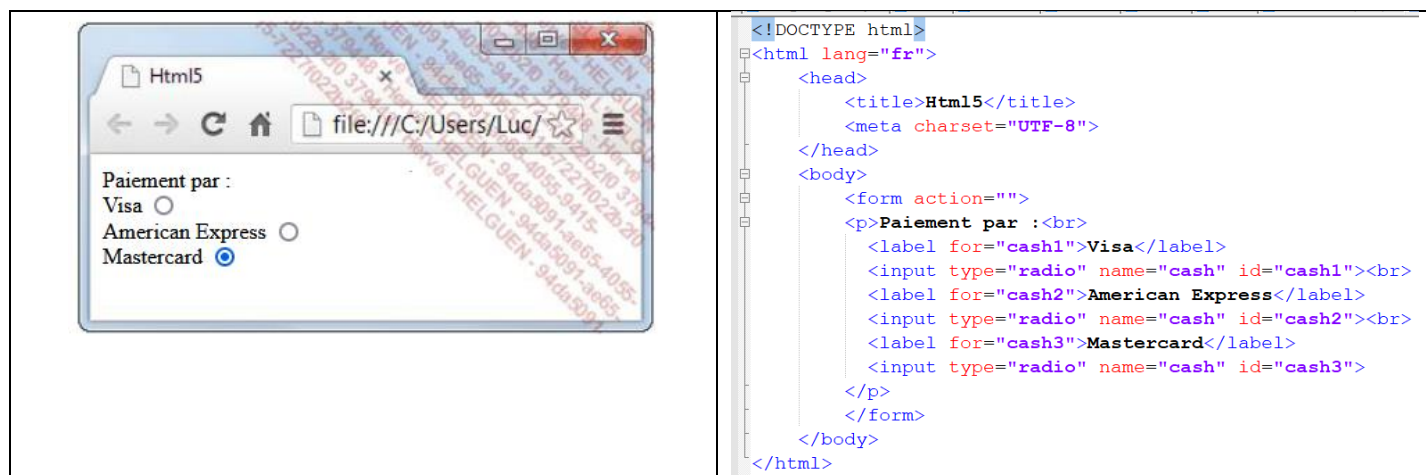
- Les groupes

Permet de regrouper des composants de façon logique dans le formulaire



- Les labels

Permet d'améliorer l'ergonomie du formulaire en mettant des étiquettes devant les composants.



## 9) Application

Créer le code HTML qui permet d'obtenir la page suivante :

The diagram illustrates a web form with two main sections: **Client** and **Commande**. The form includes various input fields, a text area, a dropdown menu, a range slider, and a spin button, each with specific HTML5 constraints and naming conventions.

**Client Section:**

- Nom et prénom :** Input field with constraint **Suggestion** and naming **txtNom**.
- Adresse :** Input field with constraint **Obligatoire** and naming **txtAdresse**.
- Code Postal :** Input field with constraint **Obligatoire** and naming **txtCP**.
- Pays :** Input field with constraint **Obligatoire** and naming **txtPays**.
- Message pour la commande :** Text area with constraint **Contrôle de saisie regex CP français** and naming **txtMessage**.

**Commande Section:**

- Modele :** Dropdown menu with constraint **Choix parmi** and naming **cboModele**. The list includes: -Road 66, -Rangers, -Cowboy, -Country.
- Taille :** Range slider with constraint **de 1 à 4 (par défaut 2)** and naming **cursTaille**. The scale is S, M, L, XL.
- Quantité :** Spin button with constraint **de 1 à 6 (par défaut 1)** and naming **valQuantite**.

**Buttons:** Commander, Recommencer, Afficher la commande.

Afin vérifier votre code vous pouvez le soumettre au site <https://validator.w3.org>

## 10) Quelques notions de JavaScript

Soit le formulaire créé dans la partie précédente dans lequel on a rajouté un bouton pour afficher la commande :

*Client*

Nom et prénom : Zappa Frank

Adresse : 17 rue de l'invention

Code Postal : 67000

Pays : France

*Commande*

Modèle : Road 66

Taille :

S M L XL

Quantité : 2

Commander Recommencer Afficher la commande

Si on clique sur **Afficher la commande**, voilà le résultat obtenu :



Pour réaliser cette action il faut passer par un langage de programmation qui s'appelle le JavaScript. Voici ce qui a été mis en œuvre.

Le script suivant a été placé dans la balise **<head>** de la page html :

```
<head>
  <title>Html5</title>
  <meta charset="UTF-8">
  <script type="text/javascript">
    function afficher() {
      var nom = document.getElementById("nom").value;
      alert(nom);
    }
  </script>
</head>
```

...et fait référence à la ligne de texte placée dans le formulaire

```
Nom et prénom : <input type="text" name="nom" id="nom" required  
placeholder="Nom et prénom"><br>
```

```
var nom = document.getElementById("nom").value;
```

Permet d'affecter à une variable nom la valeur saisie par l'utilisateur dans la zone de texte dont l'id est "nom" (l'attribut id est donc à renseigner obligatoirement en html pour l'exploiter en JavaScript).

```
alert(nom);
```

Permet d'afficher le contenu de la variable nom

## 11) Exercice

Afficher un récapitulatif de la commande ergonomique en utilisant cette technique.