1 Problématique

Une action indispensable pour un site marchand est de pouvoir recueillir des informations de la part de ses clients. Il doit par exemple récupérer l'adresse de l'acheteur pour pouvoir lui livrer ses achats. Cette prise d'informations se réalise par le biais de *formulaires*.

Comment mettre en place un formulaire sur un site web ?

2 Création du formulaire

2.1 Éléments de base

Les formulaires HTML permettent à l'utilisateur d'envoyer des données au site web. Ils sont donc composés a minima d'un *champ* que l'utilisateur devra remplir et d'un *bouton d'envoi* pour transmettre les données. La balise HTML *form* permet de créer un formulaire. Chaque champ sera crée avec une balise *input*.

Activité 1 :

1. Créer une page web puis insérer le code 1 entre les balises *body*.

```
<form>
   <input type="text">
   <input type="password">
   <input type="submit" value="Envoyer">
</form>
```

Code 1 – Premier formulaire

2. Visualiser la page web dans un navigateur.
3. Écrire dans chacun des deux champs et observer les différences. Comment les expliquer ?

Il existe de nombreux types de champs prédéfinis. La page https://tinyurl.com/yb45a39q recense les possibilités.

2.2 Soigner l'esthétique

Comme pour tout élément HTML, il est possible d'habiller un formulaire avec une page CSS. Le code 2 englobe le champ *input* dans une balise *div* à laquelle on applique une classe CSS. De plus une étiquette (*label*) est liée à ce champ.

```
<div class="form-securite">
   <label for="nom">Identifiant </label>
   <input type="text" id="nom">
</div>
```

Code 2 – Habillage d'un champ du formulaire

Christophe Viroulaud

Activité 2 :

1. Télécharger le dossier compressé *activite2-identification.zip* sur le site https://cviroulaud.github.io.
2. Dans le dossier *activite2-identification* observer le code de la page *identification.html* et de la feuille CSS associée (le fichier *traitement.php* ne sera pas utilisé dans cette activité).
3. Quel est le rôle de l'attribut *required* ?
4. En s'appuyant sur le modèle précédent, créer le formulaire (figure 1) d'un site marchand.

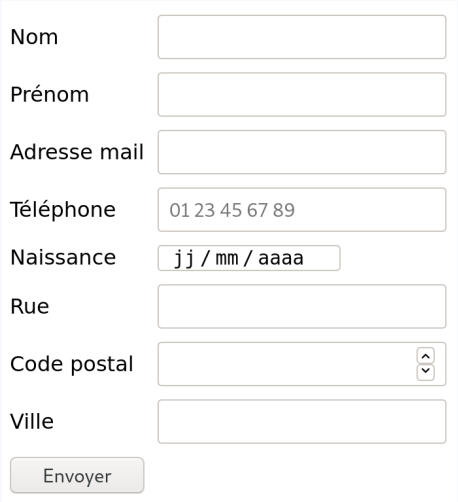


FIGURE 1 – Page de livraison

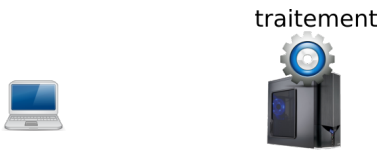
3 Envoi des données

3.1 Principe de fonctionnement

Pour l'instant le fait d'appuyer sur le bouton de validation ne provoque rien (à part recharger la page). Que doit-il se passer quand un formulaire est validé ?

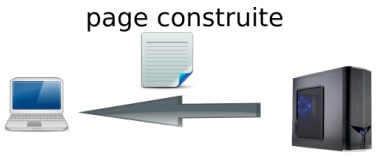
— Les données sont envoyées au serveur qui héberge le site web.

— Un fichier (souvent en langage *PHP : acronyme récursif pour PHP Hypertext Preprocessor*) traite les données. Il peut stocker les informations dans une base de données, construire une page web personnalisée en fonction des informations...



— Le serveur renvoie une page web qui utilise éventuellement les données du formulaire.

Christophe Viroulaud



Ce fonctionnement implique plusieurs conditions :

— Il est nécessaire de disposer d'un serveur distant pour pouvoir lui envoyer les données.

— Le formulaire doit être légèrement modifié pour qu'il sache quelle action réaliser.

Activité 3 :

1. Se rendre sur le site https://tinyurl.com/y78f4jmp . Ce serveur distant remplira la première condition.
2. Observer le *code source* de la page web affichée. Quels sont les éléments ajoutés par rapport à l'activité 2 ?

À retenir

— L'attribut *action* de la balise *form* cible le fichier qui s'occupera du traitement des données.

— Les champs *input* doivent contenir un attribut *name*. C'est lui qui sera utilisé par le fichier de traitement.

3.2 Méthodes d'envoi

Il existe deux méthodes d'envoi des données : GET et POST.

3.2.1 GET

Activité 4 :

1. Sur le site web précédent, compléter le formulaire avec des informations fictives, puis valider. Le site renvoie une page web qui utilise les informations.
2. Observer la barre d'adresse du navigateur. Que peut-on dire à propos de la sécurité de cette méthode d'envoi ?

À retenir

La méthode *GET* ajoute les données à l'URL. C'est la méthode d'envoi par défaut quand rien n'est précisé dans le formulaire.
Ce procédé simple peut poser des problèmes de sécurité.

3.2.2 POST

Il faut préciser la méthode d'envoi avec l'attribut *method* (code 3).

```
1 <form action="traitement.php" method="post">
```


Christophe Viroulaud

Code 3 – Méthode d'envoi

Les données sont envoyés au serveur dans le corps du message.

Activité 5 :

1. Se rendre à l'adresse https://tinyurl.com/y8qxp97p.
2. Observer le code source de la page.
3. Remplir le formulaire avec des données fictives et valider.
4. Dans le dossier *activite2-identification*, observer le code de la page traitement.php

Remarque

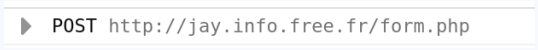
L'apprentissage du langage PHP ne fait pas partie du programme de première. La page *traitement.php* est donnée à titre d'observation.

À retenir

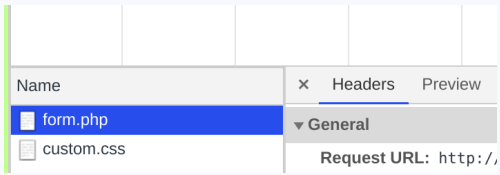
La méthode *POST* doit être précisé dans l'entête du formulaire. Les données sont envoyées dans le corps du message. Il n'y a donc pas de taille limite.
L'utilisation de cette méthode ne signifie pas que le message est sécurisé : en effet, le contenu circule en clair sur internet. S'il est intercepté, il peut être lu sans difficulté.

Activité 6 :

1. Se rendre à nouveau à l'adresse https://tinyurl.com/y8qxp97p.
2. Entrer un identifiant et un mot de passe fictifs. *Ne pas valider tout de suite.*
   Choisir le raccourci en fonction du navigateur :
3. **Sur Firefox :**
   (a) Cliquer sur *Ctrl+Shift+E*, le panneau *réseau* s'ouvre.
   (b) Dans la page web, valider le formulaire.
   (c) Ouvrir la ligne *POST*.



   (d) Dans *Requêtes*, retrouver alors les informations transmises.
4. **Sur Chrome :**
   (a) Cliquer sur *Ctrl+Shift+I*, un panneau s'ouvre.
   (b) Choisir *Réseau* ou *Network*.
   (c) Dans la page web, valider le formulaire.
   (d) Cliquer sur *form.php*.



   (e) Dans *Headers*, retrouver alors les informations transmises.

Christophe Viroulaud