

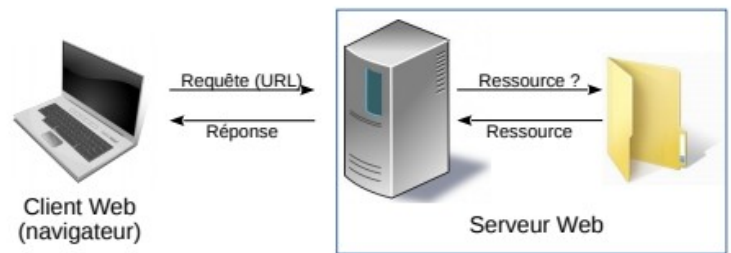
CH7 INTERFACE WEB ET FORMULAIRES

Un serveur web est une application spécifique qui tourne sur un ordinateur. Il existe des serveurs écrits en langage C, en Java, en Python,...

Le rôle du serveur Web est de servir des clients. Ces derniers envoient au serveur une requête précisant leur demande. Le serveur répond à cette demande si celle-ci est valide.

Pour communiquer le serveur et le client utilise un langage commun, il s'agit du protocole **HTTP**.

Les échanges entre le client et le serveur se font sous forme de **paquets** qui transportent sur le réseau Internet les requêtes du client et les réponses du serveur.



1. L'interface web côté client

1.1. Le formulaire HTML

Une requête est souvent plus complexe : il ne s'agit pas seulement de demander une page web, mais aussi de préciser l'état de certains paramètres pour son affichage et/ou accéder à des données particulières (base de données). Pour satisfaire ces requêtes, l'utilisateur doit saisir sa demande dans une interface que l'on appelle **formulaire**.

Exemples :

- Voir <https://polytech.univ-nantes.fr/formulaire-de-contact/>, on voit apparaître un formulaire assez classique pour Ouvrir l'inspecteur web (*Développement web* > *Inspecteur* ou bien touche *F12*) et accéder au code HTML.
- « Rendu » de certains formulaires

Contact

Nom *

Prénom

Nom

E-mail *

Message*

Envoyer

formulaire de contact type wordpress

i MODIFIER UNE BIBLIOTHÈQUE

Valider Retour

Bibliothèque Lieu Adresse Informations Territoire Configuration

Configuration du compte Redmine

Adresse

Code postal

ville *

Téléphone

Mail

Mail suggestions

Valider Retour

formulaire d'accès à une DB

Un formulaire web proposé par un client web (navigateur) permet de récupérer un grand nombre d'information saisies par l'utilisateur. Il se présente sous forme de champs à compléter (texte, date, nombre, ...), de boutons radios (choix), de « checkbox » (cases à cocher), ainsi que de boutons pour déclencher des actions spécifiques.

Un formulaire est une **IHM** (Interface Homme Machine) web qui permet à l'utilisateur de saisir sa demande, qui sera envoyée au serveur (requête).

(d'après https://fr.wikipedia.org/wiki/Formulaire#En_informatique)

→ ACT 1 Etude d'un formulaire (partie 1)

Les formulaires sont délimités par la balise **<form>**. Elle permet de regrouper plusieurs éléments de saisie utilisateur (boutons, champs, ...) et qui possède les attributs suivants :

- Attribut **method** indique sous quelle forme seront envoyées les réponses « POST » est la valeur qui correspond à un envoi de données stockées dans le corps de la requête, tandis que « GET » correspond à un envoi des données codées dans l'URL, et séparées de l'adresse du script par un point d'interrogation.
- Attribut **action** indique l'adresse d'envoi des données une fois le formulaire validé : URL, script CGI, dresse email (<mailto:adresse.email@machine>), etc...

Exemples :

- Une recherche web peut s'écrire
<https://duckduckgo.com/?t=canonical&q=cours+d%27informatique+pour+les+profs+nuls&ia=web> .

Les données sont « en clair » dans l'URL, donc avec la methode GET.

- Exemple de syntaxe : `<form action="page.php" method="get">`

Ce formulaire va envoyer les données en GET vers le fichier **page.php** sur le serveur.

Il est possible d'insérer n'importe quel élément HTML de base dans une balise *form* (textes,boutons,tableaux,liens,...) mais il est surtout intéressant d'insérer des éléments interactifs.

La balise **<input type=" " name=" " >** permet d'accéder à un ensemble de boutons et de champs de saisie en modifiant la valeur de l'attribut **type**. L'attribut **name** est la variable dans laquelle est stockée la valeur du champ de saisie.

Les deux grandes familles sont :

- Le champ de texte **<input type="text">** : une zone de saisie de texte
- Le bouton radio **<input type="radio">** : une valeur parmi plusieurs

La validation se fait en général par un bouton final **<button type="submit">... </button>**.

Pour plus d'éléments : https://www.w3schools.com/html/html_forms.asp

Exercice 1 :

Créer un formulaire de 3 champs et un bouton qui donne le visuel ci-contre :

Nom :

Prénom :

Age :

Exercice 2 :

On souhaite créer un formulaire qui affiche la couleur correspondante aux valeurs de RVB saisies.

Il faut donc 3 champs de saisie étiquetés « R », « V » et « B », un bouton « Affichage ».

Pour l'affichage de la couleur demandée, on prévoira un cadre de 100 px (balise **<div>** avec **id ="couleur"**).

R :

V :

B :

1.2. Interactions sur le formulaire

Dans les formulaires, l'envoi de paramètres se fait uniquement par une requête HTTP, et donc implique une communication en réseau avec le site distant et le serveur doit recalculer entièrement la page de destination.

L'utilisateur voit donc une nouvelle page se charger, ce qui peut être long !

L'interpréteur JavaScript d'un navigateur offre la possibilité de modifier dynamiquement et en temps réel le rendu d'une page Web, sans passer par le serveur.

Le langage JavaScript est inséré directement dans le code HTML à l'aide de la balise `<script>` :

- soit directement comme dans l'exemple ci-dessus
- soit en intégrant un lien vers un fichier .js

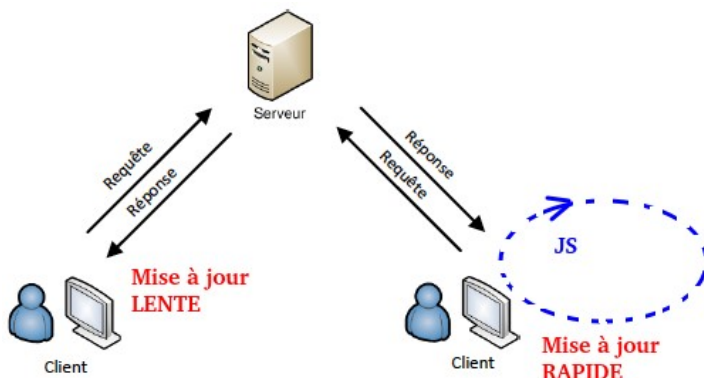
```
<script type="text/javascript" src="js/monscript.js" async></script>
```

Exemple : A partir du fichier **exemple_js1.html** donner quelques caractéristiques du langage JavaScript

```
<!DOCTYPE html>
<html>
<head><title>Age</title>
<script type="text/javascript">
let compteur = 0 ;
function suivant() {
compteur = compteur + 1 ;
document.getElementById("valeur").innerHTML = compteur;
}
</script>
</head>
<body>
<button onclick="suivant();">Clic!</button>
<span id="valeur">0</span>
</body>
</html>
```

Des instructions se comportent comme des print en python, il s'agit de `alert("message")` qui ouvre un popup avec le message et `console.log("message")` qui écrit dans la console du navigateur.

Ces instructions sont très pratiques pour voir ce que renvoie le code JavaScript dans une phase de conception.



A l'affichage d'une page, les scripts JavaScript présents sont exécutés. L'une de leur activité principale est la définition de *gestionnaires d'événements*. Ces **gestionnaires d'événements** peuvent modifier la page en ajoutant, supprimant ou modifiant les attributs des balises.

La gestion des événements par JavaScript permet donc une mise à jour plus rapide de la page (local).

Exemple : A partir du fichier **exemple_js2.html**, ouvrir le navigateur (Firefox) et appuyer sur F12 pour faire apparaître la console.

- Écrire les instructions suivantes dans la console et observer les effets :

Instruction 1	<code>document.body.style.background = "red";</code>	
Instruction 2	<code>document.body.style.background = "";</code>	
Instructions 3	<code>let d = document.getElementById("mondiv"); d.style.color = "blue"; d.style.fontWeight = "bold"; d.style.border = "1pt dashed green";</code>	
Instruction 4	<code>d.innerHTML = "Salut, <i>ça va bien ?</i>";</code>	

- Le fichier html est-il modifié ?

→ **ACT 2 Côté Client en JS**

TRAVAIL A RENDRE : MiniProjet6 Etape 2

2. Transmettre les données

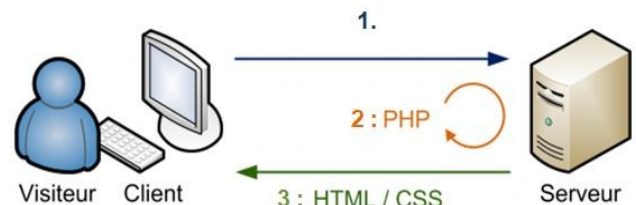
Nous avons vu que le protocole HTTP permet à des pages web d'envoyer des paramètres de requêtes. Ces derniers sont utilisés pour calculer le contenu de la page renvoyée au client. Le langage HTML permet de décrire un formulaire permettant à l'utilisateur de saisir des valeurs et de les soumettre à un serveur web.

Sur ce serveur le traitement de l'information va se faire en **langage PHP**.

PHP (pour **P**HP **H**ypertext **P**reprocessor) est un langage **libre** et **gratuit**, spécialement conçu pour le développement d'applications web. Il peut être intégré facilement au HTML.

Lorsque le serveur envoie la page web au client, il envoie en fait du code en langage HTML et CSS généré par PHP.

PHP est un langage que seuls les serveurs comprennent et qui **permet de « générer » des pages web**.



Exemple :

- Certains sites parviennent à afficher le pseudo sur toutes les pages. Étant donné que le serveur génère une page à chaque fois qu'on lui en demande une, il peut la personnaliser en fonction des données fournies par le visiteur.

Pour que son ordinateur puisse lire du PHP, il faut qu'il se comporte comme un **serveur**.

Il suffit simplement d'installer les mêmes programmes que ceux que l'on trouve sur les serveurs qui délivrent les sites web aux internautes.

Exemple :

- **Apache** (= serveur web) Il s'agit du plus important (en nombre) des serveurs utilisés sur le web
- **Python CGI** Il s'agit de la bibliothèque python qui permet de créer un serveur, simple et efficace. (voir ACT serveur python)
- Les serveurs dépendent du protocole utilisé et sont spécialisés :
HTTP pour le web, SMTP pour les mails, FTP ou SSH pour les fichiers, etc...

→ **ACT 3 Côté Serveur en PHP**

TRAVAIL A RENDRE : MiniProjet6 FINAL