

# table des matière

1. [table des matière](#)
2. [Génération de XML en Php](#)
  1. [Objectif : des pages à charger et à rafraichir par morceaux](#)
  2. [Le php en un clin d'oeil](#)
    1. [le php](#)
    2. [le php et le passage de paramètre](#)
  3. [Demonstration par l'exemple](#)
    1. [le résultat final](#)
    2. [la génération de XML](#)
    3. [l'exploitation du XML](#)
  4. [Exercices possibles](#)

## Génération de XML en Php

### Objectif : des pages à charger et à rafraichir par morceaux

A travers les requêtes asynchrones sur des fichiers XML en javascript (AJAX), il est possible d'obtenir des informations une fois la page chargée. Pour l'instant nous n'avons vu que sur des fichiers XML statiques, c'est-à-dire écrit "à la main" et non changeant.

Or, pour une plus grande dynamique du web, il serait bien de pouvoir avoir une adaptation du contenu de ces fichiers... Il faut donc des fichiers XML qui ne soient pas maintenus par le webmestre "à la main".

Une génération mécanique peut être fournie par des outils de mise en ligne (CMS ou autres logiciels) lors de la mise à jour du contenu (flux rss par exemple). Les fichiers XML peuvent aussi être générés à la volée, par exemple en Php, c'est que nous allons voir.

### Le php en un clin d'oeil

#### le php

Le php est entièrement défini sur [les pages php.net](#).

#### le php et les BDs

la connexion à une BD se fait par des fonctions propres au SGBD connecté. Pour un serveur mysql :

```
<?php
$connexion = @mysql_connect('nom_du_serveur', 'nom_de_login', 'mot_de_passe');
$dbd = @mysql_select_db( 'nom_de_la_base', $connexion);

?>
```

Ensuite, il s'agit de faire des requêtes SQL avec : `mysql_query`, comme par exemple : `$requete = @mysql_query("SELECT * FROM magasin);`.

Finalement il faut analyser le retour : vrai ou faux pour une requête de mise à jour, ou de d'insertion. Ou les réponses à une requête `SELECT` avec une boucle : .

```
while ($resultat = mysql_fetch_array($requete))
{
    // traitement de la ligne resultat
    echo "\n\t<achat>\n\t\t<produit>{$resultat["produit"]}</produit>";
    // l'indexation peut être numérique (ordre des champs) ou
    // associative (nom des champs)
}
```

## le php et le passage de paramètre

Les paramètres passés à une page web sont disponibles, s'ils existent, dans les variables "super globales" `$_REQUEST` et `$_POST` ou `$_GET`. Ce sont des tableaux associatifs. L'index est le nom (attribut "name") de la balise HTML utilisée (ou pour la méthode get le mot avant le signe égal).

## Demonstration par l'exemple

Nous allons voir un exemple de "gestion" de caddie. Il s'agit juste de mettre à jour sans recharger la page, par des requêtes asynchrones.

### le résultat final

Le résultat final est visible là : [la page fonctionnelle](#) et [les sources de cette page](#).

### la génération de XML

Pour obtenir un caddie : [la page fonctionnelle GET qui ne marche qu'avec des paramètres](#) et [les sources de cette page](#).

Pour ajouter un élément dans un caddie : [la page fonctionnelle SET qui ne marche qu'avec des paramètres](#) et [les sources de cette page](#).

## **l'exploitation du XML**

Outre la version finale (c.f. plus haut), une première exploitation est visible sur [la page testVoirCaddie.html](#).

Cette page utilise les scripts suivants :

- [une script d'utilitaire pour manipuler le contenu textuel d'une balise html](#)
- [le script qui permet d'obtenir un caddie, sans mise en forme](#). Notons qu'il utilise la programmation objet pour pouvoir faire plusieurs requêtes.

Finalement, pour effectuer des requêtes de mise à jour du caddie dans la BD, les pages utilisent [le script pour passer ou enlever des commandes](#).

## **Exercices possibles**

...

