

Un livre d'or

Par Mathieu Nebra (Mateo21)



www.openclassrooms.com

*Licence Creative Commons 6 2.0
Dernière mise à jour le 17/05/2013*

Sommaire


Sommaire	2
Un livre d'or	3
Réalisation du livre d'or	3
Étape 1 : prérequis	3
Étape 2 : préparation du script	3
Étape 3 : à vous de jouer !	5
Étape 4 : correction	5
Étape 5 : améliorez ce script !	8
Partager	9

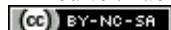


Un livre d'or

Par  Mathieu Nebra (Mateo21)

Mise à jour : 17/05/2013

Difficulté : Facile 



16 visites depuis 7 jours, classé 6/807

De nombreux sites web aiment bien démarrer avec un livre d'or pour que les membres laissent leur avis. C'est un bon moyen de récolter les premiers retours de ses visiteurs.

On va dans ce TP apprendre à créer un livre d'or pour votre site. Ainsi, les visiteurs pourront laisser une trace de leur passage, et ça valorisera d'autant plus votre site 😊

Réalisation du livre d'or

Étape 1 : prérequis

Les prérequis pour le livre d'or ressemblent beaucoup à ceux du Mini-Chat :

- savoir lire dans une table MySQL ;
- savoir écrire dans une table MySQL ;
- savoir compter le nombre d'entrées dans une table ;
- savoir bien utiliser les formulaires.

Ce que vous avez appris dans le chapitre précédent sur les formulaires est très important. N'hésitez pas à vous y reporter, car c'est en combinant les formulaires et MySQL que vous créerez des scripts très puissants !

Étape 2 : préparation du script

Comme d'habitude, on n'attaque **jamais** un script PHP avant de l'avoir préparé : on doit réfléchir à son fonctionnement si on ne veut pas se retrouver complètement perdus dans notre code.

On doit d'abord se demander quelles seront les fonctionnalités de notre livre d'or. Quand un visiteur va arriver dessus, que va-t-il pouvoir faire ? Il peut laisser son pseudo et un message ; il sera enregistré dans une table comme tous les autres.

Commençons par créer cette table justement. On va l'appeler "livreor", et elle aura 3 champs :

- **id** : comme d'habitude, on crée un id pour numéroter chaque champ. Cet id a le type INT dans MySQL et j'ai sélectionné "Primaire" comme je vous avais dit de faire pour les champs id. Par ailleurs, j'ai choisi "auto increment" pour que le numéro d'id s'inscrive automatiquement à chaque nouvelle entrée.
- **pseudo** : c'est le pseudo du visiteur. Il est de type VARCHAR, et il ne faut pas oublier de préciser sa "taille" lorsque vous créez la table : une taille de 255 me semble suffisante (vous n'avez pas des pseudos de 256 caractères dites-moi ? o_O)
- **message** : on y stocke le message qu'a laissé le visiteur. Ce champ est de type TEXT.

Je propose qu'on mette en haut de la page le formulaire, puis juste en-dessous la liste des messages... Ce qui devrait vous faire très fortement penser au Mini-Chat !

Voici un aperçu de ce que vous devez pouvoir réaliser :

Mon site vous plaît ? Laissez-moi un message !

Pseudo :

Message :

Page : [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#)

Pokity a écrit :
Je voulais remercier tous les createur du site car ce site est génial. Et je voulais savoir si vous connaissait un hebergeur de site gratuit qui accepte le php. merci est ke le site continu.

niclao2000 a écrit :
vraiment super bien foutu vot'site avec un design qui insite à revenir
bonne continuation...

mon site <http://perso.wanadoo.fr/niclao2000/>

Encore une fois, c'est moche je vous l'accorde. Mais le design, c'est très facile à réaliser (une couleur de fond, une image et c'est bon 😊).

Nous, ce qui nous intéresse c'est d'arriver à faire marcher le code. Vous aurez tout le loisir de rajouter des fioritures après, croyez-moi. 😊

Pour l'enregistrement, c'est comme pour le Mini-Chat. Par mesure de sécurité, il ne faut pas oublier le `htmlspecialchars` ainsi que le `mysql_real_escape_string` (car on va utiliser les variables dans des requêtes SQL).

Bon, vous avez certainement remarqué la principale différence entre le Mini-Chat et le livre d'or : ici on ne veut pas afficher juste les 10 derniers messages, on veut TOUS les afficher. Et tous sur une même page, ça fait beaucoup. Donc, *on va créer automatiquement des pages* pour qu'il y ait par exemple 20 messages par page.

Cela peut vous sembler flou : comment "créer" automatiquement des pages en PHP ? Vous allez voir, ce n'est peut-être pas ce à quoi vous vous attendiez. 😊

1. On récupère le nombre total des messages (`$totalDesMessages`) grâce à la requête MySQL que je vous ai apprise.
2. On calcule le nombre de pages qu'il y aura. Comment ? Avec une bête division mathématique !

$$\text{\$nombreDePages} = \text{\$totalDesMessages} / \text{\$nombreDeMessagesParPage};$$
 Exemple : si on a 100 messages, et qu'on veut 20 messages par page. On calcule $100 / 20 = 5$ pages ! Tout bête n'est-ce pas. 😊



Vous devez, bien sûr, donner d'abord une valeur à `\$nombreDeMessagesParPage`. Moi je mettrai 20 par exemple, mais vous mettez ce que vous voulez. L'avantage, c'est que si vous voulez changer plus tard le nombre de message par page, vous aurez juste à modifier la valeur de la variable !

Mais... mais... s'il y a 102 messages par exemple, $102 / 20 = 5.1$ pages ! Or, on ne peut pas avoir 5.1 pages, donc il va falloir créer une sixième page qui ne contiendra que 2 messages.

Il y a une fonction PHP mathématique qui va bien nous aider : `ceil`. Elle renvoie le nombre entier supérieur : si on lui

donne 5.1, elle va renvoyer 6 ! C'est exactement ce qu'il nous faut ! Donc, pour être sûrs du nombre exact de pages à créer, il faudra plutôt utiliser l'instruction suivante :

```
$nombreDePages = ceil($totalDesMessages / $nombreDeMessagesParPage);
```

- Maintenant qu'on a le nombre de pages, on va écrire tous les liens vers chacune de ces pages (1 2 3 4 5 6...). Lorsqu'on cliquera sur un de ces numéros, ça amènera à la page correspondante.

Il suffit de faire une boucle For (ou While, comme vous voulez :) qui va écrire tous ces nombres à la suite, de 1 à \$nombreDePages. Comme ça, on aura écrit 1 2 3 4 si on a 4 pages dans notre livre d'or !

- Mais vers quelle page amène chacun de ces liens ? En fait, on va rouvrir la même page, mais avec un paramètre différent. On va rajouter un ?page=4 par exemple si on veut aller à la page 4. Si la page du livre d'or s'appelle "livreor.php", le lien vers la page 4 sera donc :

```
<a href="livreor.php?page=4">4</a>
```

Si je prends l'aperçu du livre d'or que je vous ai montré tout à l'heure, voici par exemple les liens vers les pages 3 et 8 :

Page : 1 2 3 4 5 6 7 8

➔ ``
➔ ``

C'est ce qu'on cherche à obtenir.

Ça a peut-être l'air d'être un truc mathématique barbare, mais dites-vous bien que ce n'est qu'une minable petite division à faire. Non, en fait ce qui doit vous gêner, c'est que vous découvrirez le "truc" des pages pour la première fois.

On ne va pas créer un fichier PHP différent pour chaque page (je suis pas fou à ce point 😊). Simplement, il y aura une variable \$_GET['page'] qui va indiquer à quelle page on se trouve. Notre script saura afficher les messages de cette page tout seul !



Pensez à vérifier si \$_GET['page'] existe. Si ce n'est pas le cas, alors c'est la première fois que le visiteur charge le livre d'or : mettez-le sur la page 1 (la plus récente).



Bon, j'ai récupéré le nombre total de messages et j'ai écrit les liens vers chacune de ces pages avec une boucle. Je fais comment pour afficher juste les messages de la page ?

Souvenez-vous : \$_GET['page'] est une variable qui contient le numéro de la page que le visiteur est en train de lire. Il suffit de modifier la requête SQL, en utilisant LIMIT. Et n'oubliez pas qu'on veut afficher les messages dans l'ordre décroissant (le plus récent en premier).

Il va donc falloir réfléchir un petit peu à la requête SQL... Je ne vous donne pas la réponse, il faut que vous cherchiez vous-mêmes.

Pour info, j'ai bien dû réfléchir 5 minutes avant de trouver la requête... Eh oui, contrairement aux apparences je ne suis pas un robot : comme vous, je dois réfléchir un peu avant d'arriver à faire marcher mes scripts. 😊

Bon, et après ça, c'est gagné. 😊

Il ne vous reste plus qu'à afficher le résultat de la requête dans une boucle, comme vous avez l'habitude de faire. Si tout se passe bien, les messages de la page s'afficheront !

Étape 3 : à vous de jouer !

Pfiou ! Je vous ai décortiqué tout le problème, vous avez les bases qu'il faut pour écrire le script. 😊

Prenez votre temps surtout, on n'est pas pressés.

Soignez bien le système de pages, c'est d'ailleurs la seule chose qui différencie ce TP du TP Mini-Chat.

Surtout, n'abandonnez pas trop vite : il est parfois bien, quand vous commencez à avoir mal à la tête, d'aller piquer un petit somme. Très souvent, on a plein de bonnes idées au réveil ! 😊

Ah, et n'ayez pas peur de créer d'autres variables si nécessaire. C'est un bon moyen de simplifier un problème, parfois !

Étape 4 : correction

On corrige maintenant ?

Allez, voici en gros ce qu'il fallait faire :

Code : PHP

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
  <head>
    <title>Livre d'or</title>
    <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />

    <style type="text/css">
      form, .pages
      {
        text-align:center;
      }
    </style>
  </head>
  <body>

    <form method="post" action="livreor.php">
      <p>Mon site vous plaît ? Laissez-moi un message !</p>
      <p>
        Pseudo : <input name="pseudo" /><br />
        Message :<br />
        <textarea name="message" rows="8"
cols="35"></textarea><br />
        <input type="submit" value="Envoyer" />
      </p>
    </form>

    <p class="pages">

<?php
mysql_connect("localhost", "sdz", "mot_de_passe");
mysql_select_db("coursphp");

// ----- Étape 1 -----
// Si un message est envoyé, on l'enregistre
// -----

if (isset($_POST['pseudo']) AND isset($_POST['message']))
{
    $pseudo =
mysql_real_escape_string(htmlspecialchars($_POST['pseudo'])); // On
utilise mysql_real_escape_string et htmlspecialchars par mesure de
sécurité
    $message =
mysql_real_escape_string(htmlspecialchars($_POST['message'])); // De
même pour le message
    $message = nl2br($message); // Pour le message, comme on
utilise un textarea, il faut remplacer les Entrées par des <br />

    // On peut enfin enregistrer :o)
    mysql_query("INSERT INTO livreor VALUES('', '' . $pseudo . '',
'' . $message . '')");
}

// ----- Étape 2 -----
// On écrit les liens vers chacune des pages
// -----

// On met dans une variable le nombre de messages qu'on veut par
page

```

```

$nombreDeMessagesParPage = 20; // Essayez de changer ce nombre pour
voir :)
// On récupère le nombre total de messages
$retour = mysql_query('SELECT COUNT(*) AS nb_messages FROM
livreor');
$donnees = mysql_fetch_array($retour);
$totalDesMessages = $donnees['nb_messages'];
// On calcule le nombre de pages à créer
$nombreDePages = ceil($totalDesMessages /
$nombreDeMessagesParPage);
// Puis on fait une boucle pour écrire les liens vers chacune des
pages
echo 'Page : ';
for ($i = 1 ; $i <= $nombreDePages ; $i++)
{
    echo '<a href="livreor.php?page=' . $i . '>' . $i . '</a> ';
}
?>

</p>

<?php

// ----- Étape 3 -----
// Maintenant, on va afficher les messages
// -----

if (isset($_GET['page']))
{
    $page = $_GET['page']; // On récupère le numéro de la page
indiqué dans l'adresse (livreor.php?page=4)
}
else // La variable n'existe pas, c'est la première fois qu'on
charge la page
{
    $page = 1; // On se met sur la page 1 (par défaut)
}

// On calcule le numéro du premier message qu'on prend pour le
LIMIT de MySQL
$premierMessageAfficher = ($page - 1) * $nombreDeMessagesParPage;

$reponse = mysql_query('SELECT * FROM livreor ORDER BY id DESC LIMIT
' . $premierMessageAfficher . ', ' . $nombreDeMessagesParPage);

while ($donnees = mysql_fetch_array($reponse))
{
    echo '<p><strong>' . $donnees['pseudo'] . '</strong> a écrit
:<br />' . $donnees['message'] . '</p>';
}

mysql_close(); // On n'oublie pas de fermer la connexion à MySQL
;0)
?>

</body>
</html>

```

Veillez noter : j'ai volontairement désactivé l'ajout de messages sur cet exemple de livre d'or afin d'éviter que ça ne devienne un bazar sans nom. 😞

Vous pouvez donc lire les messages mais pas en ajouter. Testez ce script en local avec WAMP, et vous verrez qu'il marche bien entendu. 😞

Bien entendu, je ne doute pas que 90 % d'entre vous ont eu du mal à arriver à finir ce script (ou ont bloqué quelque part).

Je ne vous le répèterai jamais assez : il ne faut pas s'affoler, c'est normal. Le TP vous force à réfléchir, et la correction vous permet de vous dire : « Ah oui ! C'était ça ! »

C'est ça qui est vraiment enrichissant pour vous. 😊

Et si nous expliquions un peu ce code ?

Comme vous pouvez le voir, il est clairement séparé en 3 parties :

1. **Partie 1 : Si un message est envoyé, on l'enregistre** : comme d'habitude, on vérifie si les variables `$_POST['pseudo']` et `$_POST['message']` existent. Si c'est le cas, c'est que le visiteur vient d'envoyer un message. On fait alors un `mysql_real_escape_string` et un `htmlspecialchars` sur chacune de ces variables pour éviter que du HTML ne soit inscrit dans votre livre d'or. On utilise aussi un `nl2br` pour `$_POST['message']` : en effet, si le visiteur a tapé des Entrées dans le textarea, il faut utiliser `nl2br` comme je vous l'ai appris pour les transformer en balises `
`. Après, il n'y a plus qu'à envoyer une requête à MySQL pour qu'il enregistre le message dans la table.
2. **Partie 2 : On écrit les liens vers chacune des pages** : bon là ça devient un peu plus chaud. Je vous ai pas mal expliqué auparavant comment il fallait faire, je vais rapidement vous rappeler ce qu'il se passe.
 1. On définit une variable `$nombreDeMessagesParPage`. J'ai choisi 20 messages par page. L'avantage d'utiliser une variable pour retenir le nombre de messages par pages est évident : si vous décidez plus tard de changer le nombre de messages par page, vous aurez juste à changer cette ligne et non pas tout le code. 😊
 2. On récupère le nombre de message total avec une requête SQL comme on l'a vu dans la partie II. On met ce nombre dans `$totalDesMessages`
 3. Puis, on calcule le nombre de pages à créer avec la division dont je vous ai parlé. Le ceil permet, je vous le rappelle, d'avoir un nombre de pages entier (et non pas 5.2 pages 😊).
 4. Enfin, on n'a plus qu'à faire une boucle pour écrire tous les numéros de page (et les liens qui vont avec). J'ai choisi de faire une boucle `for` car c'est pratique dans ce cas, mais si vous avez fait un `while` c'est tout aussi bon.
3. **Partie 3 : Maintenant, on va afficher les messages** : il reste une dernière petite difficulté, celle-là je vous avais laissé la résoudre vous-mêmes (sinon autant vous donner la correction de suite 😊). On a 3 choses à faire :
 1. On vérifie si `$_GET['page']` existe (avec un `isset`). Si c'est le cas, on place ce numéro de page dans `$page`. Si ce n'est pas le cas, c'est que l'adresse de la page est "livreor.php" (il n'y a pas de numéro de page indiqué dans l'adresse). Alors, on donne la valeur 1 à `$page`.
 2. Maintenant qu'on a un bon numéro de page dans `$page`, on doit calculer quelque chose pour la requête. En effet, on va faire un `ORDER BY id DESC LIMIT ?, ?`
Que mettre dans chacun de ces points d'interrogation ?
 - Le premier point d'interrogation correspond au numéro du premier message à prendre (qui n'a rien à voir avec l'id). Le premier message a le numéro 0 je vous rappelle. Comme on prend les messages dans l'ordre décroissant, on commence par le dernier. Si on est sur la page 1, $(\$page - 1) * \$nombreDeMessagesParPage$ va renvoyer $(1-1) * 20 = 0 * 20 = 0$. On prendra donc à partir du message n°0 (le premier quoi), et comme on est dans l'ordre décroissant, ça correspond au tout dernier message enregistré. Donc, sur la page 1, on affichera en premier le dernier message écrit dans le livre d'or : exactement ce qu'on voulait ! 😊
 - Pour le second point d'interrogation, c'est facile : il s'agit du nombre d'entrées à récupérer dans la table, c'est-à-dire `$nombreDeMessagesParPage` puisqu'on veut prendre juste les X messages de la page.
 3. Ouf, le plus dur est fait. On n'a plus qu'à faire la requête SQL pour prendre juste les messages qu'il faut, puis à faire une boucle pour afficher les messages de la page.

Eh oui, ce script était un peu plus gros que les autres, d'où l'intérêt de bien le découper en plusieurs parties !

Prenez le temps de bien comprendre comment ce système de pagination fonctionne, croyez-moi ça vaut le coup et vous en aurez certainement besoin pour un de vos futurs scripts. 😊

Étape 5 : améliorez ce script !

Sur un tel script, on pourrait faire beaucoup d'autres choses. Je vais vous proposer quelques pistes pour améliorer ce script, mais je ne vous donnerai pas de correction (sinon je m'en sortirai jamais 😊).

Je pense que c'est une bonne idée que vous continuiez à réfléchir à l'amélioration du script tout seuls. Mon rôle ici se limite à vous donner des idées.

Et n'oubliez pas que je ne vous abandonne pas : si vous bloquez sur une amélioration vous pouvez aller demander de l'aide sur les forums !

- Améliorez le design, c'est vraiment super basique ce que j'ai fait comme présentation. C'est facile à faire, ça demande juste de modifier le code HTML.
- Affichez le nombre total de messages postés dans votre livre d'or, c'est pratique de le voir écrit (c'est super facile à faire,

si vous n'y arrivez pas c'est très grave 😊).

- Ajoutez une liste déroulante dans votre formulaire. Ce champ permettra au visiteur de choisir une note qu'il donne à votre site (mettez donc les choix 0 1 2 3 ... 18 19 20).

Ajoutez un champ "note" de type INT à votre table livreor (si vous regardez dans les options de PHPMyAdmin c'est facile de rajouter un champ même quand la table a été créée).

Comme ça, en plus du message du visiteur, vous affichez la note qu'il a donnée à votre site

- Mieux, vous faites la moyenne de toutes les notes qu'on a données à votre site, et vous l'affichez en haut (ex. : "Note moyenne : 14.6 / 20"). Je vous rappelle que pour trouver la moyenne, il faut faire $(note1 + note2 + note3 + \dots) / totalDesMessages$
- Question sécurité, si on veut vraiment être parfaits, il y a un petit détail que j'ai omis volontairement pour pas trop compliquer le script. Vous voyez le numéro de page passé par l'adresse avec `$_GET['page']` ? Eh bien, si le visiteur modifie manuellement ce numéro de page, il peut mettre n'importe quoi et même du code HTML !

Bon, dans ce cas c'est pas bien grave, le code HTML ne sera pas affiché et ça fera juste bugger le script. Mais, pour être "propres", il faudrait utiliser la fonction `intval` sur `$_GET['page']` pour transformer à coup sûr le numéro de page en un nombre. Donc, il faudrait écrire plutôt la ligne suivante :

```
$page = intval($_GET['page']);
```

Si vous ne le faites pas vous n'allez pas en mourir, mais sachez que les bons codeurs en PHP font ce genre de choses. Je peux pas vous en vouloir de ne pas l'avoir fait, vous ne connaissiez pas cette fonction. 🤖

Voilà ce que j'appelle un bon vrai TP. 😊

Là, on est vraiment dans le concret des choses : vous êtes aptes à créer des scripts assez importants, comme vous venez de le voir. Le truc pour pas se perdre, c'est une organisation sans faille : un code bien présenté, avec des commentaires et des noms de variables clairs.

Vous n'allez pas me croire, mais c'est la clé de tout. Oui oui, un code bien présenté vous permet de ne pas vous noyer, des commentaires permettent de vous y retrouver.

Prenez ces bonnes habitudes, il va sans dire qu'elles seront tout bonnement indispensables si vous voulez bien progresser. 😊

Partager



Ce tutoriel a été corrigé par les [zCorrecteurs](#).