

CC PHP : Session 1 - 1h30 - Sur machine

T l chargez l'archive **cc-18-php-sujet.tgz** de l'espace Moodle. Apr s extraction, renommez le r pertoire obtenu **nom-prenom** o  **nom** et **prenom** sont vos nom et pr nom  crits en minuscules et sans espaces (par ex. **delafontaine-jean**). A l'issue de l'examen, archivez ce r pertoire (`tar czf nom-prenom.tgz nom-prenom`) et d posez l'archive sur Moodle.

Vous conserverez la structure du r pertoire **nom-prenom** qui est organis  en 3 sous-r pertoires :

- **data** : contient le script SQL pour reconstituer la base de donn es.
- **src** : contient diff rents fichiers que vous devrez compl ter (  l'exception de **js.php**, **qcm.html** et **qcm.css**).
- **test** : contient diff rents scripts de tests que vous pourrez ex cuter avec PHP en ligne de commande ou via le navigateur.

Les exercices portent sur la r alisation de pages web permettant aux visiteurs de r pondre   diff rents QCM portant sur le langage PHP. Ces pages s'appuient sur une base de donn es qui contient 40 questions et qui associe   chacune plusieurs r ponses dont une seule est correcte.

Le visiteur renseigne d'abord son nom sur la page d'accueil (figure 1) et y choisit un nombre n de questions. S'il ne coche pas la case Tirage al atoire, les n premi res questions de la base sont s lectionn es et affich es au clic du bouton D marrer (figure 2). Dans le cas contraire, les n questions sont tir es al atoirement (figures 5 et 6).

Le visiteur r pond ensuite aux questions en cochant une r ponse par question (figure 3). Apr s soumission, le QCM est r affich  : les questions apparaissent en vert ou rouge selon que le visiteur a r pondu correctement ou non et toutes les solutions sont coch es (figure 4).

Le visiteur peut alors retenter ce QCM (bouton Recommencer !) ou revenir   la page d'accueil (bouton Accueil !) pour g n rer un nouveau QCM.

Votre nom

Nombre de questions

Tirage aléatoire ☐

FIGURE 1 – Page d’accueil

Que fait l'instruction `include('file.php');` si `file.php` n'existe pas ?

☒ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☐ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

Que fait l'instruction `require('file.php');` si `file.php` n'existe pas ?

☒ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☐ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

`$x = null; echo gettype($x);`

☒ NULL
☐ boolean
☐ undefined

FIGURE 2 – Affichage du QCM généré

Que fait l'instruction `include('file.php');` si `file.php` n'existe pas ?

☐ Rien.
☒ Une erreur de niveau `E_WARNING` est émise.
☐ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

Que fait l'instruction `require('file.php');` si `file.php` n'existe pas ?

☒ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☐ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

`$x = null; echo gettype($x);`

☐ NULL
☐ boolean
☒ undefined

FIGURE 3 – Choix des réponses

Que fait l'instruction `include('file.php');` si `file.php` n'existe pas ?

☐ Rien.
☒ Une erreur de niveau `E_WARNING` est émise.
☐ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

Que fait l'instruction `require('file.php');` si `file.php` n'existe pas ?

☐ Rien.
☐ Une erreur de niveau `E_WARNING` est émise.
☒ Une erreur fatale de niveau `E_COMPILE_ERROR` est émise.

`$x = null; echo gettype($x);`

☒ NULL
☐ boolean
☐ undefined

[Recommencer ! Accueil !](#)

FIGURE 4 – Résultat

Exercice 1. Créez sous phpMyAdmin une base de données MySQL de nom `l3info_cc_18_autoqcm` en choisissant le moteur InnoDB et le jeu d'interclassement `utf8mb4_bin`. Importez y le script `l3info_cc_18_autoqcm.sql`. L'import crée la table `QUESTION` qui stocke les questions, la table `ALTERNATIVE` qui stocke les réponses proposées pour chaque question, et crée plusieurs enregistrements pour chaque table. `QUESTION` est dotée des colonnes suivantes :

- `ID` : l'identifiant de la question (`int(3)`, `AUTO_INCREMENT`, `PRIMARY_KEY`)
- `ENONCE` : l'énoncé de la question (`varchar(500)`).

`ALTERNATIVE` est dotée des colonnes suivantes :

- `ID_QUESTION` : l'identifiant de la question associée (clé étrangère)
- `REPONSE` : l'énoncé de la réponse (`varchar(500)`, `NOT NULL`)
- `SOLUTION` : pseudo-boléen indiquant si la réponse est la solution à la question (valeur 1 le cas échéant, 0 sinon) (`tinyint(1)`, `NOT NULL`).

Exercice 2. Complétez le script `connexpdo.inc.php` afin d'utiliser le compte `root` qui vous est attribué par le conteneur Docker pour se connecter au SGBD MySQL. Testez le script en exécutant `test-connexpdo.php`.

Exercice 3. La fonction `selectionnerQuestions` du script `selectionner-questions.php` sera utilisée pour sélectionner et extraire sous forme de tableau l'énoncé des questions qui seront à afficher, qu'elles soient tirées aléatoirement ou non. Implémentez cette fonction en vous conformant aux commentaires fournis dans le fichier. Testez-la en exécutant `test-selectionner-questions.php` : la figure 7 illustre le résultat attendu.

Votre nom

Nombre de questions

Tirage aléatoire ☒

FIGURE 5 – Tirage aléatoire

```
$tab=[]; array_push($tab[0],$tab[0]); print_r($tab);
```

☒ Array ()

☐ Array([0] => Array ([0] => Array ()))

☐ Array([0] => [])

☐ Une erreur de niveau E_WARNING est émise.

FIGURE 6 – QCM généré aléatoirement

```
=====
Tirage aléatoire de 2 questions :
Array
(
    [3] => <code>$x = null; echo gettype($x);</code>
    [10] => <code>var_dump(TRUE==1);</code>
)

=====
Sélection des questions 11 et 26 :
Array
(
    [11] => <code>echo 234 <=> 123;</code>
    [26] => <code>$x='\n'; echo "$x";</code>
)

=====
Levée d'exception (1) :
$a : type incorrect
=====
Levée d'exception (2) :
$a : entier hors plage
=====
Levée d'exception (3) :
$a : entier hors plage
=====
Levée d'exception (4) :
valeur d'identifiant incorrect dans tableau $a
=====
Levée d'exception (5) :
valeur d'identifiant incorrect dans tableau $a
```

FIGURE 7 – Test de la fonction
selectionnerQuestions

```
Array
(
    [0] => Array
        (
            [REPONSE] => <code>-1</code>
            [SOLUTION] => 0
        )
    [1] => Array
        (
            [REPONSE] => <code>0</code>
            [SOLUTION] => 0
        )
    [2] => Array
        (
            [REPONSE] => <code>1</code>
            [SOLUTION] => 1
        )
)
```

FIGURE 8 – Test de la fonction
selectionnerAlternatives

Exercice 4. La fonction `selectionnerAlternatives` du script `selectionner-alternatives.php` sera utilisée pour extraire sous forme de tableau les réponses associées à une question donnée. Implémentez cette fonction en vous conformant aux commentaires fournis dans le fichier. Testez-la en exécutant `test-selectionner-alternatives.php` : la figure 8 illustre le résultat attendu.

Exercice 5. La fonction `selectionnerQuestionsAlternatives` du script `selectionner-questions-alternatives.php` sera utilisée pour sélectionner et extraire sous forme de tableau l'énoncé des questions qui seront à afficher ainsi que leurs réponses. Implémentez cette fonction en vous conformant aux commentaires fournis dans le fichier et en faisant appel aux deux fonctions précédentes `selectionnerQuestions` et `selectionnerAlternatives`. Testez-la en exécutant `test-selectionner-questions-alternatives.php`.

Exercice 6. Les énoncés de questions et réponses en base de données sont des chaînes de caractères constituées de texte en français, de code PHP, ou d'un mélange des deux, par exemple :

- Que fait l'instruction `<code>include('file.php');</code>` si `<code>file.php</code>` n'existe pas ?
- Rien
- `<code>echo 234 <=> 123;</code>`

La fonction `encoderEnonce` du script `encoder-enonce.php` sera utilisée pour convertir en HTML les énoncés des questions et réponses. Précisément, elle doit convertir tous les caractères spéciaux HTML dans la chaîne mais laisser intact les balises `<code>` et `</code>` qui y figurent. Implémentez cette fonction en vous conformant aux commentaires fournis dans le fichier. Testez-la en exécutant `test-encoder-enonce.php`. Notez que les énoncés peuvent apparaître correctement dans le navigateur si les énoncés sont affichés tels quels (i.e. sans utiliser cette fonction).

Exercice 7. La soumission du formulaire de la page d'accueil (figure 1) invoque le script `generer-qcm.php`. Ce dernier importe le gabarit `gabarit-qcm.php` après avoir récupéré le tableau des questions et réponses via la fonction

`selectionnerQuestionsAlternatives`. Complétez le fichier **gabarit-qcm.php** afin de générer l’affichage des questions/réponses tel qu’illustré en figure 2. Respectez les consignes suivantes :

- placez chaque question avec ses réponses dans un élément HTML `div` ;
- placez l’énoncé de la question dans un élément `p` ;
- placez l’énoncé de chaque réponse dans un élément `label` ;
- associez à chaque réponse un bouton radio ayant pour attribut `name` l’identifiant de la question et pour attribut `value` l’indice de la réponse dans le tableau des réponses ;
- utilisez `encoderEnonce()` pour convertir les énoncés de questions et de réponses ;
- faites en sorte que la première réponse à chaque question soit cochée par défaut.

Exercice 8. La soumission des réponses choisies par l’utilisateur invoque le script **traiter-qcm.php**. Ce dernier réaffiche le QCM en indiquant toutes les solutions et en coloriant en vert et rouge respectivement les bonnes et mauvaises réponses de l’utilisateur. Complétez le fichier **traiter-qcm.php** afin de générer l’affichage du résultat tel qu’illustré en figure 4. Exploitez à cet effet les règles CSS prédéfinies dans le fichier.

Exercice 9. Après affichage des résultats (figure 4), un clic sur le bouton `Recommencer` ! doit réafficher le QCM qui vient d’être tenté par le visiteur, qu’il ait été généré aléatoirement ou non au départ. Mettez cette fonctionnalité en place à l’aide de sessions en modifiant certains des scripts PHP.