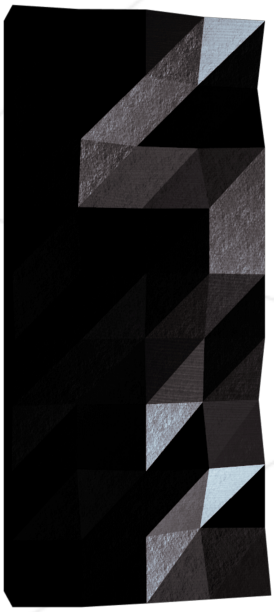

Piscine PHP

Jour 09

Staff 42 piscine@42.fr

Résumé: Ce document est le sujet du jour 09 de la piscine PHP de 42.

Table des matières

Chapitre I

Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Seul le travail rendu sur votre dépot sera pris en compte par la correction.
- L'utilisation d'une lib interdite est un cas de triche. Toute triche est sanctionnée par la note de `-42`.
- Vous ne devez laisser dans votre répertoire aucun autre fichier que ceux explicitement spécifiés par les énoncés des exercices.
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Votre manuel de référence s'appelle `Google`.
- Pensez à discuter sur le forum. La solution à votre problème s'y trouve probablement déja. Sinon, vous en serez l'instigateur.
- Réfléchissez. Par pitié, par Thor, par Odin ! Nom d'une pipe !

Chapitre II

Préambule

Gollum : Qu'est-ce qu'il fait ? Stupide Hobbit joufflu, ça les abime.

Sam : Abimer quoi ? Ils n'ont presque pas de viande. Ce qu'il nous faudrait, c'est des bonnes patates.

Gollum : C'est quoi des patates, mon précieux ? Qu'est-ce que c'est ?

Sam : Pommes de terre, idiot. En bouillie, en purée, en rondelles dans un ragoût. De bonnes grosses frites dans l'huile pour accompagner du poisson frit. Même vous n'y résisteriez pas.

Gollum : Oh si, nous y résisterions. Gâcher ainsi du bon poisson. Nous, nous le préférons cru et frétillant. Gardez vos vilaines frites.

Sam : Vous êtes désespérant.

Chapitre III

Veuillez souffler dans le ballon

	Exercice : 00
Veuillez souffler dans le ballon	
Dossier de rendu : $ex00/$	
Fichiers à rendre : `balloon.html`	
Fonctions Autorisées : `HTML, CSS, JS`	
Remarques : `n/a`	

Pour cet exercice nous allons gonfler un ballon. Pour débuter, votre ficher html aura une structure de base comme celle vue dans la vidéo d'introdution. Aucune lib n'est autorisée.

Créez en HTML/CSS un `div` de `200px` par `200px` avec en couleur de fond le rouge. Ses bordures seront arrondies afin de créer un rond parfait et non plus un carré. Ce `div` sera donc notre ballon.

Lors d'un clic sur le ballon, sa taille augmentera de `10px` tout en gardant sa forme arrondie. À chaque clic, sa couleur passera dans cet ordre du rouge, au vert, au bleu puis on reprend depuis le rouge.

En règle générale ce type de ballon est plutôt résitant mais si sa taille est supérieure à `420px`, il "explose" et reprend sa taille initiale.

Petit détail suplémentaire, lorsque le curseur de la souris est dans le ballon puis le quitte, la taille du ballon diminue de `5px` (la taille du ballon ne peut pas descendre en dessous de `200px`) et sa couleur change dans l'ordre inverse que précédemment cité.

Chapitre IV

It's over 9000

	Exercice : 01
It's over 9000	
Dossier de rendu : $ex01/$	
Fichiers à rendre : `calc.html`	
Fonctions Autorisées : `HTML, CSS, JS`	
Remarques : `n/a`	

Pour cet exercice nous allons faire une calculatrice basique. Pour débuter, votre ficher HTML aura une structure de base comme celle vue dans la vidéo d'introdution. Aucune lib n'est autorisée. Le design importe peut du moment que l'exercice reste réalisable

Tout d'abord modélisons notre calculatrice. Elle sera composée de la manière suivante :

- un `input` de type `text` qui réprésente le membre de gauche de notre calcul.
- un `select` qui contiendra en options la liste des opérateurs suivants (`'+'`, `'-'`, `'*'`, `'/'`, `'%'`).
- un `input` de type `text` qui réprésente le membre de droite de notre calcul.
- un `input` de type `submit` et de valeur `'Try me!'`.

Lorsque l'on clique sur le bouton `'Try me!'`, le calcul est déclenché puis le résultat apparaît dans une fenêtre d'alerte. Le résultat devra être également affiché sur la console de votre navigateur (log).

Le membre de gauche et de droite ne doivent contenir que des entiers positifs (>= 0) pour que le calcul puisse ce réaliser. Sinon il faut afficher une alerte avec le texte `'Error :('`.

La division et le modulo par zéro afficherons comme alerte : `"It's over 9000!"` ainsi que dans la console toujours dans les log.

Toutes les 30 secondes, une message d'alerte doit apparaître en disant `'Please, use me...'`.

Chapitre V

To do or not to do

	Exercice : 02
To do or not to do	
Dossier de rendu : $ex02/$	
Fichiers à rendre : `index.html, todo.js`	
Fonctions Autorisées : `HTML, CSS, JS`	
Remarques : `n/a`	

Pour cet exercice nous allons faire un mini gestionnaire de tâches en local. Pour débuter, votre ficher HTML aura une structure de base comme celle vue dans la vidéo d'introdution. Aucune lib n'est autorisée. Le design importe peut du moment que la structure présentée ci-dessous est respectée. Soyez créatif mais concentrez vous avant tout sur le côté fonctionnel.

La to do list sera représenté par un `div` qui aura pour attribut `id` la valeur `'ft_list'`. Ce bloc contiendra la liste des to do. Chaque to do sera représenté par un div contenu dans le bloc `'ft_list'`. Quand un to do est créé, il est placé en haut de la liste. À vous de créer l'élément et de le placer au bon endroit (manipulation de DOM).

Il faut également un bouton de création avec pour valeur d'affichage `'New'`. Ce dernier déclenche lors d'un clic, une fênetre de texte (voir fonction prompt) qui permet de renseigner le nouveau to do. Une fois le texte validé, s'il n'est pas vide, il devra apparaitre en haut de la liste.

Pour supprimer un to do de la liste, il suffit de cliquer dessus. Une fenêtre de confirmation doit s'ouvrir et demander si oui ou non vous souhaitez supprimer ce to do. Si vous confirmez, le to do doit définitivement disparaitre du DOM, il ne faut donc pas juste le "cacher".

Petite implémentation suplémentaire, votre to do list devra être sauvegardée sous forme de cookies. Si la liste contient des to do et que vous fermez le navigateur, cette même liste devra être chargée et affichée dans `'ft_list'`. Si le(s) cookie(s) n'existe(nt) pas alors la liste sera vierge.

Chapitre VI

Si jQuery, j'y vais aussi

	Exercice : XXbis
Si jQuery, j'y vais aussi	
Dossier de rendu : *exXXbis/*	
Fichiers à rendre : `Mêmes fichiers que les exercices respectifs`	
Fonctions Autorisées : `HTML, CSS, jQuery`	
Remarques : `n/a`	

Il est temps d'utiliser l'outil tant espéré. Vous devez donc refaire les 3 premiers exercices en utilisant la lib jQuery. Importez donc la lib via CDN comme présenté dans la vidéo à ce sujet. Inutile de télécharger la lib dans votre dépot sinon vous perdez des points.

Pour le répertoire de rendu, faites 3 dossiers distincts : `ex00bis/`, `ex01bis/` et `ex02bis/`

Uniquement la lib jQuery est autorisées pour ces exercices.

Chapitre VII

AJAX, nettoyant surpuissant

	Exercice : 03
AJAX, nettoyant surpuissant	
Dossier de rendu : $ex03/$	
Fichiers à rendre : `index.html, todo.js, select.php, insert.php, delete.php, list.csv`	
Fonctions Autorisées : `HTML, CSS, jQuery, PHP`	
Remarques : `n/a`	

Maintenant que vous avez un peu plus la main avec jQuery, il est grand temps de vous jeter à l'eau (de javel) ! Le stockage en local c'est bien mais dans le cloud c'est mieux.

Pour le coup votre cloud ne sera pas très loin. Il sera sur votre serveur qui executera le php en localhost.

Vous devez donc remplacer dans l'exercice précédent le système de sauvegarde pour que le tout soit fonctionnel avec une sauvegarde 'distante' via CSV et non plus via cookies. Les données dans le CSV seront sous la forme `'id;i am a todo'`.

Vous devez obligatoirement utiliser de l'AJAX. Votre page HTML ne doit jamais subir de rafraichissement.

- le fichier `select.php` récupère la liste des to do depuis le CSV.
- le fichier `insert.php` ajoute un to do dans le CSV.
- le fichier `delete.php` supprime un to do dans le CSV.

Toute action sur la page en rapport avec votre liste (ajout, suppression) devra être reportée dans le CSV par des appels AJAX.

Bon courage !