WORDPRES : Les résolutions (via julio potier)

Développer avec la constante `WP_DEBUG` sur `TRUE`.

L'avantage de cette constante est de pouvoir vous apercevoir rapidement d'une erreur de code, les warnings et notices seront alors affichées.

Utiliser les fonctions/APIs WordPress au lieu des fonctions/APIs PHP ou mes fonctions/APIs maison.

L'avantage d'utiliser le code de WordPress est que vous passerez par des hooks, ce dont raffolent les dévs !

Ajouter des actions et des filtres dans mon code.

Je viens de le dire, on en raffole, cela ne vous coûte rien d'ajouter de temps en temps une ligne de code, aucune perf de perdue, promis !

Ne pas toucher arbitrairement à des paramètres que je ne devrait pas.

N'allez pas filtrer non stop un élément du code, ne modifiez pas une options de WordPress sans demander la permission sous prétexte que votre plugin en a besoin, merci d'avance.

Commenter et aérer mon code un gros minimum.

Essayez au moins de laisser quelques espaces entre certains blocs, et commentez l'utilisation de la fonction et le code.

Utiliser des `isset(), empty(),in_array(), array_key_exists(),defined()` etc ...

Vous éviterez les warnings et notices !

Tester mes développements, et re-tester...

Utiliser les fonctions de désinfection (`sanitize_*`) et d'échappement (`esc_*`) ou de filtrage (`wp_kses*`).

Ces fonctions de sécurité permettrons à votre plugin de ne pas se retrouver parmi les plugins vulnérables en `version` 0.1b. Vous éviterez principalement la faille XSS avec ça.

Utiliser les fonctions `wp_create_nonce(),wp_nonce_url(), wp_nonce_field(),settings_fields()`,

Même chose pour la faille CSRF.

Éviter de charger des librairies kikoolol.

Parfois on a pas le choix, parfois on l'a, faite le bon !

Éviter de nommer mes fonctions de façon trop classiques.

Ne commencez jamais par `wp_` et ne soyez pas trop court ou trop classique comme `search` ou `getlist`.

Faire commencer chacun de mes fichiers plugins ou thèmes par `defined( 'ABSPATH' ) or die( 'Direct acces not allowed!' );`

Cela évite une erreur fatale puisque votre fichier utilise des fonctions WordPress, encore non déclarées lors d'un accès direct !

Arrêter de charger mes propres petites icônes, vive dashicons ou les sprites,

On est en 2015 quand même ...

Éviter de faire un appel direct à un fichier inclus dans un plugin,

Utilisez les hooks via des actions, voir `admin-post.php`.

Rendre mes développements prêts à être traduits avec `__(), _e(), _x(),_n()` etc...

Les fonctions de traductions sont disponibles, utilisez les (mais bien !).

WORDPRES : Les résolutions (suite)

Traduire les mots dans mes fichiers JavaScript au lieu de tout mettre en dur.
Oui c'est ennuyant ça, mais avec `wp_localize_script() (en)` ça va vite !

N'ajouter les styles et scripts que sur mes pages et pas partout tout le temps,
Utilisez les marqueurs conditionnels en front ou choisissez les bons hooks en back pour l'enqueue de vos CSS et JS.

Coder d'abord en NOJS puis rendre le dev compatible JS et non l'inverse,
Pensez que tout le monde n'a pas le JS actif, codez d'abord sans JS si possible.

Utiliser **`admin-post.php`** avant d'utiliser **`admin-ajax.php`**,
Même chose qu'au dessus, si vous devez faire de l'ajax, faites le d'abord via admin-post.php,

Rendre mes développements compatibles avec la plus ancienne version de PHP, MySQL et WP possible.
WordPress 4.1 tourne sous PHP 5.2.4, MySQL 5.0, pourquoi pas vos développements ?

Arrêter de recalculer à chaque itération de boucle un sélecteur jQuery.
Le fait de recalculer fait vraiment perdre des secondes au visiteurs, déclarez une variable au préalable !

Factoriser mon code pour éviter la redondance et faciliter la maintenance,
Si vous vous apercevez que plusieurs blocs ou fonctions de votre code sont quasi identiques, alors factorisez les !

Utiliser les tags HTML **`button`** au lieu des **`a`** pour faire des boutons,
Ceci est un point SEO et accessibilité, les liens a ne seront pas visités, et les boutons seront alors de vrais boutons.

Utiliser les ARIA et autres possibilités de rendre vos développements accessibles,
Il y a plus de gens qui ont besoin d'accessibilité que vous ne le pensez, et il ne faut pas être 100% handicapé visuel pour en avoir besoin.

Éviter la surqualification CSS pour améliorer les perfs clients.
Un `[type=text]` suffit contre un `input[type=text]`, idem un `#foo` au lieu d'un `div#foo`.

Toujours utiliser les braces dans les **`IF`**, **`FOR`**, **`FOREACH`**, **`WHILE`**, etc ... même pour une seule ligne de code.
Cela permets d'éviter de mettre une seconde ligne hors contexte lors d'une prochaine modification du code.

Indenter mon code avec des tabulations et non des espaces.
Cela rends le code lisible plus facilement, et les tabulations ont été choisies comme standard pour WordPress.

Utiliser **`$wpdb->prepare()`** avec au moins 1 paramètre lors des requêtes maison.
Cela éviter les failles injections SQL, voir prepare (en).

Rendre le code lisible et maintenable facilement pour les humains plutôt de gagner un nombre de lignes pour la machine.
Mieux vaut coder sur 5 lignes lisibles par un humains que réussir à le faire sur 1/2 ligne, illisible.

Ne pas mettre de chemins en dur dans mon code.
Il existe assez de fonctions et de constantes pour les chemins sous WordPress (en) et PHP pour éviter de mettre wp-content/plugins en dur.