Google Maps

Archriss a créé un site internet pour un conseil général. Sur ce site est référencé plusieurs adresses, comme ceux de clubs sportifs, ou d’association. Sur ce site, ils ont mis en place l’api Google Maps, permettant d’affiche une carte avec des épingles pointant sur les adresses stockées dans la base. Pour que les épingles pointent au point endroit il faut renseigner les coordonnées GPS. N’ayant que les adresses, ils ont créé un script permettant de récupérer les coordonnées GPS en fonction d’une adresse postal. Le script s’est bien exécuté, mais certaines coordonnées GPS sont soit erronées (ne pointant pas au bon endroit, car la même rue existe ailleurs), soit les coordonnées sont nul, car Google Maps n’a pas trouvé l’adresse indiquée. Il y a au total un peu de 1.000 adresses enregistrées dans la base de données. Bien évidemment, toutes les occurrences ne sont pas concernées par ces problèmes de coordonnées GPS, une majorité d’enregistrement sont correctes.

Mon tuteur de stage me demande alors d’essayer de remédier à ce problème. Après une discussion ensemble, nous en dédions que les coordonnées GPS égal ou inférieur à zéro sont fausses. Alors il dump toutes les occurrences des clubs sportifs remplissant le critère convenu. Et me l’envoi, pour qui je puis réaliser des tests en local sur ma machine. Nous étudions également le moyen de récupérer les coordonnées GPS d’une adresse. La saisie d’une url, en y renseignant l’adresse permet de retourner dans le format soit XML, soit Json, les informations concernant l’adresse saisie. L’exemple qui suit renvoie les informations de l’adresse 24 rue des Lilas à Paris, France, au format Json.

* http://maps.googleapis.com/maps/api/geocode/json?address=24ruedeslilas,Paris,France&sensor=false

J’injecte donc ce « dump », dans la base de données de test de « Xampp ». Et commence la programmation du script permettant de recherche les coordonnées GPS. Je me connecte à ma base de données avec la fonction native php « mysql\_connect ». Après quelques tests me permettant de m’assurer de la connexion de mon script à la base, je récupère les identifiants et adresses ayant des coordonnées erronées. Une fois ces données extraire dans la base, je rédige l’url me permettant d’avoir accès aux informations de l’adresse. L’idée étant d’utilisé CURL, pour récupérer le contenu en Json généré par l’exécution de cette URL. Il est donc primordial de bien former cette URL. Et c’est ceci qui m’a causé le plus de problème. Non pas qu’elle n’est mal composé, mais parce que les données extraite de la base, qui compose la partie importante de l’URL, comporte des caractères interdit dans une URL, comme par exemple des accents ou des espaces. Je me suis mis naïvement en quête d’une fonction permettant de remplacer la lettre avec accent par son équivalente sans accent. PHP n’intègre pas ce genre de fonction en natif, et plusieurs internautes proposent leur propre fonction, sur des forums ou des blogs, mais aucune n’a eu l’effet escompté. En effet, soit elle supprimé la lettre, et donc l’adresses n’était plus valable, soit elle le remplacer par un caractère dont l’encodage avait un problème (le caractère n’était pas reconnue lors de l’exécution dans le CURL). Et l’utilisation de la fonction « utf8\_encode » ne change rien. Même l’utilisation des fonctions native « urlencode » et « rawulrencode » ne change rien au problème (bien au contraire). Ce problème me bloque un long moment.

Mon tuteur de stage me dit alors de supprimer les accents. Je prends alors le dump qu’il m’a envoyé, et grâce à mon éditeur de texte, je procède à un « recherche/remplacer », qui me permet rapidement de supprimer tous les accents. Malgré cela, le retour du Json pour certaines occurrences, se révèle infructueuse. Mon tuteur de stage me signale que j’ai déjà bien dégrossi le travail, et que les quelques derniers qu’il reste seront fait à la main.

La suite du script va être développée plus rapidement. En effet, je récupère le Json renvoyé par le CURL, et le décode, grâce à la fonction « json\_decode », qui, si on passe en second paramètre à TRUE, retourne le Json sous forme de tableau. Je vérifie que la cellule « statuts » soit égale à « OK », ce qui signifie que l’exécution de l’URL a bien retournée les informations sur l’adresse. Si la condition est vérifiée, alors je stocke dans 2 variables, les latitudes et longitudes de l’adresse qui se trouve ['results'][0]['geometry']['location']. Je rédige la requête de mise à jour de la base, que j’ai préalablement testée dans ma base de données. Puis crée un fichier sql, avec la fonction « fopen », dans lequel je consigne la requête que je viens de faire. Puis referme le fichier. Puis je termine à fermant la ressource CURL.

Je vérifie que le fichier SQL, que j’ai généré fonction est exécuté sur ma base en local, qui fonctionne du premier coup.

Ce fut très intéressant de réaliser ce script, et j’ai constaté que les problèmes d’encodage sont complexes et parfois incompréhensible.