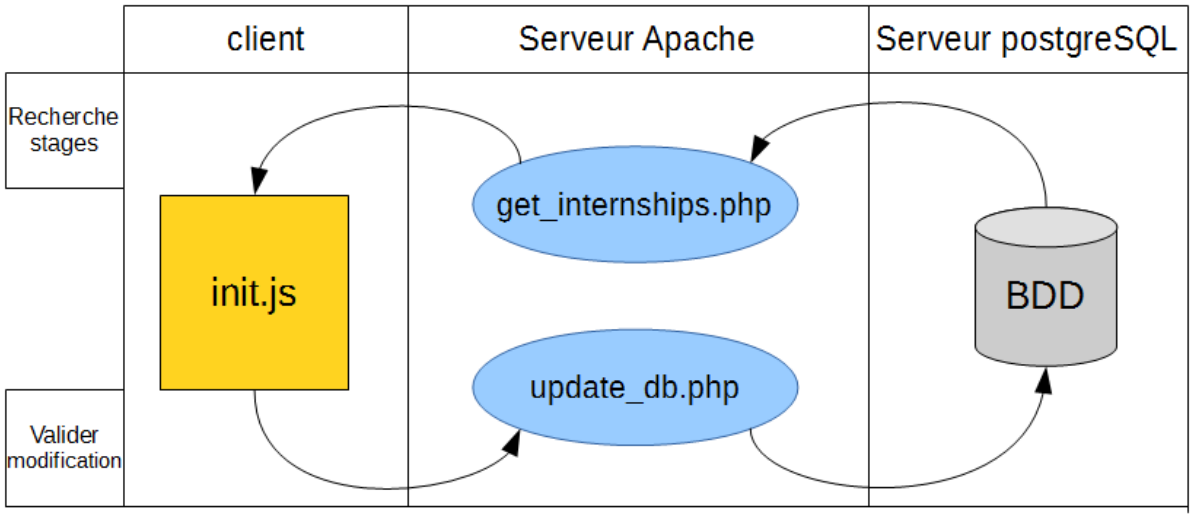
DOCUMENTATION PROGRAMMEUR

CARTE DES STAGES - INTERFACE DE SEPARATION DES SUPERPOSITIONS

FONCTIONNEMENT SCHEMATIQUE



Le client dispose de 2 fonctions, la recherche de stages et la validation de modification de géométrie.

PARTIE BASE DE DONNEES

Le serveur de base de données est de type postgreSQL, muni de postGIS.

La base de données (nommée par défaut « internships », cf. paragraphe PHP) comporte une table (par défaut « internships »), qui comporte au moins les champs suivants : lat, lng, lat_c, lng_c. Ils correspondent respectivement à la latitude et la longitude saisies par l'auteur du stage, et aux coordonnées corrigées par l'éditeur de service (ces dernières peuvent donc être vides). Il s'agit par conséquent de variables de type pgSQL « double_precision ».

PARTIE PHP

MODIFICATION DE L'ACCES A LA BASE DE DONNEES

```php
// Lien vers la base de données postGIS en local
$link = pg_connect("host=localhost port=5432 dbname=internships user=postgres password=postgres");
```

On peut modifier les paramètres d'accès à la base de données (adresse, port, nom de la base, utilisateur, mot de passe) dans les 2 fichiers PHP : à la **ligne 6** dans *get_internships.php*, et à la **ligne 8** dans *update_db.php*.

```php
// Nom de la table dans la BDD
$table_name = "internships";
```

Pour modifier le nom de la table à laquelle on accède, il suffit de modifier la variable « table_name », situé à la **ligne 6** dans *get_internships.php*, et à la **ligne 4** dans *update_db.php.get_internships.php*.

GET_INTERNSHIPS.PHP

On récupère en premier lieu les coordonnées de la recherche de stages, puis on réalise les 2 requêtes spatiales suivantes :

- On récupère les 2 stages les plus proches
- On récupère jusqu'aux 28 stages suivants les plus proches dans un rayon de 0.25° (soit environ 28km)

Le résultat envoyé au client est un JSON de tous les résultats, dans l'ordre de la distance angulaire.

UPDATE_DB.PHP

On récupère le JSON envoyé par le client contenant une liste de stages corrigés, identifées par leur ID de la base de données. On réalise une requête UPDATE dans la base de données pour chacun de ses stages. On renvoie une chaîne de caractères : « mise à jour OK ».

PARTIE CLIENT

On utilise l'API cartographique GoogleMaps, sur laquelle on a ajouté une petite surcouche permettant de gérer les marqueurs superposés (https://github.com/jawj/OverlappingMarkerSpiderfier).

INITIALISATION DE LA PAGE

La fonction *window.onload*, qui est donc lancée une fois la page chargée, est la fonction *init().* Elle fait d'abord appel à la fonction initMap(), qui initialise la GoogleMap, puis fait une première recherche de stages autour du centre de cette carte. Ensuite, elle initialises les éléments DOM et leur ajoute des écouteurs d'évènement.

```
function initMap() {
    /**
    Initalisation de la map, centrée sur l'ENSG
    */
    map = new google.maps.Map(document.getElementById('map'), {
        zoom: 12,
        center: new google.maps.LatLng(48.841014, 2.587320),
    });

    // On initialise l'instance de gestion de marqueurs superposés
    oms = new OverlappingMarkerSpiderfier(map);
}
```

On peut modifier le centre de la carte par défaut dans l'attribut *center* de la map, avec des coordonnées au format latitude, longitude.

RECHERCHE DE STAGES

La recherche de stages est effectuée par la fonction searchInternships(event), qui est aussi un écouteur d'évènement pour le bouton HTML de recherche. Elle procède par appel AJAX au fichier *get_internships.php*. Les paramètres de cette requête sont lat et lon, qui correspondent aux coordonnées du centre de la carte.

Une fois la réponse (au format JSON) disponible, on la parcourt en ajoutant les marqueurs correspondants aux résultats. Pour chaque stage, un couple de marqueurs est chargé (portant le même numéro), un marqueur rouge fixe qui correspond aux coordonnées *lat, lng* de la base de données, et un marqueur bleu mobile aux coordonnées *lat_c, lng_c*.

MISE A JOUR DE LA BDD

La mise à jour est réalisée à l'aide de la fonction sendPosition(event), écouteur d'évènement sur le bouton HTML de validation. On parcourt la liste de marqueurs bleus, et on envoie une liste d'objets (id, lat_c, lng_c) au format JSON au fichier *update_db.php*.

Lorsque le serveur répond, on alerte l'utilisateur à l'aide d'une fenêtre surgissante.