Documentation Programmeur
Antipodes - Partie Client

Ce projet a été réalisé en partie à partir du travail réalisé par Yangzhuo Peng,
sa documentation n'est ici pas répertoriée

On résume quand même légèrement ici ce que ce code nous donne :
le rectangle que vous voyez se créer lorsque vous zoomez sur une carte : c'est ça.
Les deux cartes sont bien liées et les niveaux de zoom de ces deux dernières déterminent sur quelle carte on veut "zoomer". Ainsi, cette partie crée des rectangles pour limiter l'emprise de la carte sur laquelle on veut zoomer ; vue d'une autre manière, l'emprise de la carte sur laquelle on zoom est visible grâce à ce rectangle sur l'autre carte

L'**objectif** était de programmer la partie client du projet : afficher sur chacune des deux cartes les 5 villes les plus importantes.

A savoir : deux dossiers seulement sont utiles au projet : celui du test unitaire qui fonctionne rapidement puisque l'on actualise pratiquement directement les bons marqueurs dans la bonne emprise grâce aux villes dans "antipodes.json" ; et celui produit lorsque l'on a mis en commun la partie client et la partie serveur, qui est logiquement moins rapide mais qui semble aussi fonctionnel.

Dans ces deux scripts, on retrouve dans l'idée :

- changeBoundaries() une fonction qui sert dans le cas où j'ai besoin d'envoyer mon emprise à une page php qui établit une connexion avec la base de données (qui contient une centaine de milliers de villes). Selon plusieurs critères (niveaux de zoom ou sur l'emprise en question), je détermine alors si j'envoie l'emprise d'un rectangle, d'une map et dans les deux cas, pour afficher les villes sur quelles maps.

- addLabels() puisqu'une fois que je sais sur quelles emprises je veux avoir les villes, il ne me reste plus alors qu'à fetch
    - soit le json qui sort toutes les villes, puis j'ai une fonction getTopFiveCities() pour me sortir les 5 villes les plus peuplées dans l'emprise
    - soit le php pour directement me sortir ces 5 villes.

  ensuite dans les deux cas, il faut bien placer les marqueurs pour chaque ville, ce qui est fait avec des forEach()

Pour gérer l'affichage dès le rafraîchissement de la page, on a soit une technique de compteurs dans addLabels() pour le json et directement addFirstLabels() pour la BDD.

Ensuite, pour gérer l'affichage des différents marqueurs, j'ai créé des L.featureGroup() que j'ajoute ensuite à la carte que je veux selon si c'est dans l'emprise ou dans l'emprise aux antipodes. Puis je clearLayers() à chaque actualisation pour tous les faire disparaître avant de retrouver les villes dans la nouvelle emprise donnée.

Pour plus de détails, la documentation me semble fournie et plus concrète.
A noter : la documentation de map_integre.js doit être plus étoffée et doit permettre une meilleure compréhension que map.js si vous souhaitez commencer à lire la documentation.